

UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR



PROYECTO FIN DE CARRERA

RECONOCIMIENTO FACIAL EN TIEMPO REAL

Ingeniería de Telecomunicación

Javier Eslava Ríos

Julio 2013

RECONOCIMIENTO FACIAL EN TIEMPO REAL

AUTOR: Javier Eslava Ríos
TUTOR: Pedro Tomé González

Área de Tratamiento de Voz y Señales
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio 2013

Resumen

Resumen

En este proyecto se estudia, implementa y evalúa un sistema automático de reconocimiento facial en tiempo real. Para llevarlo a cabo, se han utilizado diversas técnicas del estado del arte y estudiado su aplicación a sistemas de vídeo aprovechando así las ventajas que los mismos pueden ofrecer.

Como punto de partida se ha realizado un estudio de los sistemas comerciales y las técnicas de reconocimiento facial existentes en el estado del arte seleccionando aquellos funcionales para su aplicación a sistemas en tiempo real. Se han diseñado e implementado todas las etapas que componen a un sistema automático de reconocimiento facial con el objetivo de comparar aspectos relevantes como la precisión y el coste computacional.

El diseño del proyecto ha seguido dos líneas principales de trabajo organizadas en el tiempo, primero se implementó un sistema off-line de reconocimiento facial basado en librerías OpenCV/C++, cuyo rendimiento fue evaluado con diversas bases de datos: BANCA y FOCS, de libre acceso a la comunidad científica. Posteriormente se adaptó y mejoró el diseño e implementación para su funcionamiento en tiempo real (on-line). Estos sistemas han sido diseñados siguiendo una estructura modular de forma que cada etapa pueda ser modificada o ampliada fácilmente.

Para la parte experimental se han llevado a cabo experimentos diferenciados de cada una de las etapas del sistema, de modo que podamos evaluar el sistema desarrollado de forma detallada con sus puntos fuertes y débiles. Los experimentos se han centrado en el estudio de las etapas más relevantes para el rendimiento: detección facial, extracción de características y clasificación. Para dichas etapas se llevaron a cabo experimentos evaluando las tasas error y los tiempos de procesamiento vinculados a las mismas. Por último se ha evaluado el rendimiento global del sistema desarrollado comparándolo con sistemas de referencia del estado del arte.

El proyecto culmina con la implementación de un sistema final (demostrador/prototipo con interfaz gráfica de control) totalmente funcional disponible en el laboratorio de investigación donde se ha desarrollado dicho proyecto. Generando así una herramienta educativa que servirá de base para la investigación y desarrollo de nuevas técnicas y desarrollos aplicados al reconocimiento facial en tiempo real y a la videovigilancia.

Palabras Clave

Sistema biométrico, reconocimiento facial en tiempo real, eigenfaces, extracción de características.

Abstract

In this project, an automatic real time face recognition system is studied, implemented and evaluated. To accomplish this, various state of the art techniques have been used and their applications in video systems studied in order to take advantage of the benefits they may offer.

First of all, the commercial systems and state of the art face recognition techniques are studied in order to select the ones capable of functioning under real time applications. All the phases that form an automatic facial recognition system have been designed and implemented with the goal of comparing relevant aspects such as precision and computational cost

The design of the project has followed two main areas of work organized in time. First of all, an off-line face recognition system based in OpenCV/C++ libraries was implemented. The performance of said system, was evaluated with two databases: BANCA and FOCS, freely available to the scientific community. Afterwards, the design was adapted and enhanced for real time use (on-line). Those systems have been designed following a modular structure that easily allows modifications and extensions of each phase.

For the experimental part, tests have been carried out for each phase of the system so it can be evaluated extracting the strong and weak points. The experiments have been focused in the study of the most relevant phases in terms of performance; facial detection, feature extraction and classification. Experiments were carried out for the mentioned phases evaluating error rates and processing times bound to them. Lastly, the global performance of the system has been evaluated and compared to state of the art reference systems.

The project finishes with the implementation of a final, fully functional system (demo/prototype with graphic user interface for control) available at the research laboratory where the project has been developed. This generates an educational tool that serves as the basis for research and development on new techniques as well as development applied to real time facial recognition and video surveillance.

Key words

Biometric system, real time face recognition, eigenfaces, feature extraction.

Agradecimientos

Me gustaría agradecer en primer lugar la dedicación durante todo el proyecto de mi tutor Pedro Tomé. Su esfuerzo y su capacidad para abordar todo tipo de problemas es digno de admirar sobre todo en esos momentos cuando tiene una gran cantidad de trabajo a sus espaldas y sin embargo nunca te niega su atención cuando necesitas su ayuda. Gracias además por tu apoyo en el resto de tareas que hemos realizado aparte del proyecto y por esa gran organización para cumplir con todos los plazos.

A Javier Ortega y a Julián Fierrez me gustaría también agradecer especialmente la oportunidad que me brindaron de poder contribuir al grupo y de poder formarme en otras áreas al mismo tiempo que realizaba el proyecto. Javier, gracias por la confianza que depositaste en mí para abordar las tareas del congreso, fue una etapa en la que tuve la oportunidad de aprender muchas cosas trabajando contigo. Julián, te agradezco tu apoyo y preocupación por mis estudios tanto dentro del grupo como por los futuros.

Gracias al grupo ATVS puesto que durante mi relativamente corta pero intensa estancia, me he sentido en todo momento apoyado y ayudado por todos. Todas las personas que lo componen contribuyen al buen ambiente que hay y siempre están dispuestos a echar una mano con lo que sea. Me gustaría mencionar especialmente a Ester González por su gran ayuda durante el congreso. Ester gracias a tu ayuda pudimos hacernos con el congreso y me evitaste una alopecia prematura. Ahora ya podemos mirar atrás y reirnos de los momentos en que teníamos pesadillas con las facturas. Sin olvidarme de la inestimable ayuda de Sara, Fátima, Sandra y María que realizaron una gran organización de todos los recursos y personas así como de la ayuda de Alicia la cual tenía una particular obsesión con cierta caja, nunca sabremos por qué. A todos vosotros atvsianos, muchas gracias por este año.

Por último agradezco a mi familia su apoyo durante toda la carrera y la confianza que siempre han tenido en mí. Gracias por hacer posible que pueda perseguir mis metas y por apoyarme incondicionalmente en cualquiera de mis decisiones de futuro, todo esto me motiva a seguir esforzándome. Gracias también a mis amigos y compañeros por los buenos ratos que te hacen recordar que no todo se reduce a los estudios, sobre todo a Sergio y Maya, compañeros de días interminables en el laboratorio ¡Ánimo que ya no os queda nada!. Gracias a ti Leyre, porque eres la que literalmente ha hecho que esto sea posible.

*Javier Eslava Ríos
Julio 2013*

*Nunca olvido una cara
pero con la suya haré una excepción.*

Groucho Marx

Índice general

Índice de figuras	VII
Índice de tablas	X
1. Introducción.	1
1.1. Motivación del proyecto.	1
1.2. Objetivos y enfoque.	2
1.3. Metodología y plan de trabajo.	2
1.4. Organización de la memoria.	4
1.5. Contribuciones.	4
2. Estado del arte.	7
2.1. Introducción al reconocimiento facial en entornos reales	7
2.2. Historia del reconocimiento facial	8
2.3. Fundamentos del reconocimineto facial.	9
2.4. Ventajas e inconvenientes del reconocimiento facial.	10
2.5. Aplicaciones de los sistemas de reconocimiento facial.	11
2.6. Etapas de un sistema de reconocimiento facial.	12
2.6.1. Adquisición de la imagen.	13
2.6.2. Detección.	13
2.6.3. Preprocesado y normalización.	14
2.6.4. Extracción de características.	15
2.6.5. Comparación y Reconocimiento.	18
2.7. Sistemas comerciales y SDKs.	20
2.7.1. Sistemas comerciales en la actualidad.	20
2.7.2. SDKs de reconocimiento facial.	25
3. Sistemas desarrollados.	29
3.1. Sistema off-line	29
3.1.1. Descripción	29

3.1.2.	Entrenamiento e inicialización del sistema	31
3.1.3.	Detección	31
3.1.4.	Preprocesado (Normalización)	34
3.1.5.	Extracción de características	36
3.1.6.	Clasificación y decisión	39
3.2.	Sistema en tiempo real	40
3.2.1.	Descripción	40
3.2.2.	Equipamiento utilizado	42
3.2.3.	Adaptación e implementación del sistema off-line para su funcionamiento en tiempo real	42
3.2.4.	Desarrollo de la interfaz	44
3.2.5.	Generación de logs e información del sistema	50
4.	Experimentos realizados y resultados.	53
4.1.	Bases de datos y protocolo experimental	53
4.1.1.	Base de datos: BANCA	53
4.1.2.	Base de datos: FOCS	54
4.2.	Resultados experimentales	55
4.2.1.	Resultados de detección (Exp.Det)	56
4.2.2.	Resultados de reconocimiento (Exp.Rec)	62
4.2.3.	Resultados de los tiempos de procesamiento de las etapas (Exp.Time) . .	66
5.	Conclusiones y trabajo futuro	71
5.1.	Conclusiones.	71
5.2.	Trabajo futuro	72
	Glosario de acrónimos	75
	Bibliografía	76
	A. Presupuesto	81
	B. Pliego de condiciones	83
	C. Anexo C: manuales de utilización.	87
C.1.	Sistema off-line	87
C.2.	Sistema en tiempo real	88
	D. Anexo D: manual del programador	91

Índice de figuras

1.1. Diagrama del plan de trabajo seguido.	3
2.1. Sistema de reconocimiento facial.	7
2.2. La cara como rasgo biométrico.	9
2.3. Sistema de reconocimiento facial en los juegos olímpicos de Pekín.	11
2.4. Identificación de individuos durante los disturbios de Londres.	12
2.5. Diagrama general de un sistema de reconocimiento facial.	13
2.6. Características tipo Haar.	14
2.7. Detección de la región de la cara.	14
2.8. Preprocesamiento de una imagen. a) Imagen de entrada. b) Rotación a partir de las coordenadas de los ojos. c) Recorte y escalado en base al estándar ISO/IEC 19794-5. d) Ecualización del histograma.	15
2.9. Taxonomía del reconocimiento facial.	16
2.10. Eigenfaces obtenidas a partir de la proyección de las imágenes en el espacio PCA.	16
2.11. Vectores de distancias obtenidos a partir de puntos característicos.	17
2.12. LBP sobre imágenes con diferente intensidad.	18
2.13. Cálculo del plano que mejor divide las dos clases en un problema de dos dimensiones.	19
2.14. Empresas desarrolladoras de sistemas de reconocimiento facial.	20
2.15. Next Identification Programme.	21
2.16. SmartGate implantado en aeropuertos australianos.	22
2.17. MorphoFace Investigate.	22
2.18. Toshiba Face Recognition.	23
2.19. Entrenamiento de Face Unlock.	24
2.20. Google Glass.	24
2.21. Extracción de puntos característicos con Perceptual Computing SDK.	25
2.22. Entrenamiento con verilook SDK.	26
3.1. Diagrama de estados del sistema off-line.	30
3.2. Ejemplos de a) imagen de test y b) imagen de entrenamiento.	31

3.3. Ejemplo de detección facial usando el algoritmo Viola-Jones.	32
3.4. Región acotada para la detección de ojos.	33
3.5. Ejemplos de detección de ojos mediante Viola-Jones.	34
3.6. a) Binarización correcta. b) Binarización propensa a fallos.	34
3.7. Imágenes normalizadas.	35
3.8. Rotación de la imagen.	35
3.9. Punto de selección de 200 componentes principales una vez entrenado el PCA. . .	36
3.10. Imagen a) antes y b) después de ecualizar su histograma.	37
3.11. Operador LBP de radio 1.	38
3.12. Imágenes transformadas mediante el operador LBP.	38
3.13. Cálculo de los histogramas LBP de cada región.	39
3.14. Diagrama de etapas del sistema en tiempo real.	41
3.15. Equipamiento completo montado en el laboratorio.	43
3.16. Interfaz del programa de reconocimiento en tiempo real.	46
3.17. Sistema funcionando en modo identificación.	47
3.18. Detalle del sistema identificando correctamente a un usuario.	48
3.19. Sistema detectando múltiples caras.	49
3.20. Ejemplo de imágenes capturadas y procesadas para entrenar el sistema.	50
3.21. Diagrama de estados del funcionamiento de la comprobación de identificadores. .	50
4.1. Muestra de imágenes en la base de datos BANCA. a) <i>controlled</i> b) <i>degraded</i> c) <i>adverse</i>	54
4.2. Muestra de imágenes en la base de datos FOCS. a) <i>good</i> b) <i>bad</i> c) <i>ugly</i>	55
4.3. Usuarios con fallos en la detección de cara en los grupos a) <i>controlled</i> , b) <i>degraded</i> y <i>adverse</i>	58
4.4. Ejemplos de detección correcta e incorrecta para a) <i>Controlled</i> b) <i>Degraded</i> c) <i>Adverse</i>	59
4.5. Ejemplos de detección correcta e incorrecta para a) <i>Good</i> b) <i>Bad</i> c) <i>Ugly</i>	59
4.6. Comparación de técnicas en a) <i>Controlled</i> b) <i>Degraded</i> c) <i>Adverse</i>	61
4.7. Comparación de técnicas en a) <i>Good</i> b) <i>Bad</i> c) <i>Ugly</i>	62
4.8. Curva ROC para las técnicas implementadas en BANCA a) grupo <i>g1</i> b) grupo <i>g2</i> . 63	
4.9. Curvas ROC para las técnicas implementadas en los grupos a) <i>good</i> b) <i>bad</i> y c) <i>ugly</i> de FOCS.	65
4.10. Comparación de los resultados del sistema <i>baseline</i> respecto a los obtenidos con el sistema <i>system2mod</i>	66
4.11. Diagrama de etapas del sistema en tiempo real de las cuales se extraen los tiempos de procesamiento.	67

4.12. Tiempos de proceso para a) la detección de una cara y b) detección múltiple. . .	68
4.13. Diagrama de tiempos para las etapas involucradas en el modo entrenamiento. . .	69

Índice de tablas

4.1. Datos sobre la base de datos BANCA.	53
4.2. Datos sobre la base de datos FOCS.	54
4.3. Resultados de la detección de cara en BANCA.	56
4.4. Resultados de la detección de cara en FOCS.	57
4.5. Resultados de la detección de ojos en BANCA.	60
4.6. Resultados de la detección de ojos en FOCS.	61
4.7. Comparación de tasas de error en varios sistemas que utilizan LBP (junto con dos métodos del estado del arte) y los implementados. Tabla extraída de [1].	64
4.8. Medidas de tiempos (ms) para cada etapa del sistema (los resultados pueden tener un error de $\pm 5ms$).	67
4.9. Medidas de tiempos (ms) para cada etapa del sistema en modo entrenamiento.	69

1

Introducción.

1.1. Motivación del proyecto.

Las necesidades de mejorar la seguridad en diferentes contextos de la sociedad actualmente, dan lugar al desarrollo de varias técnicas de reconocimiento automático basadas en rasgos biométricos. Los sistemas de biometría permiten la identificación de individuos de forma segura y rápida, puesto que no se precisa mayor información que la contenida en dichos rasgos inherentes a las personas. Es por ello que estos sistemas suponen una ventaja frente a sistemas tradicionales que suelen requerir información adicional como documentos de identidad, tarjetas de acceso, etc. que puede ser perdida o robada provocando brechas de seguridad.

El gran número de campos de aplicación de técnicas biométricas hace de las mismas un objeto de estudio muy interesante para la comunidad científica. Junto con la mejora significativa de diferentes sistemas biométricos en los últimos años y la imparable expansión y evolución de la tecnología, algunos sistemas se empiezan a imponer como una solución robusta y fiable de identificación con tasas de reconocimiento muy fiables [2]. Estos sistemas son capaces de mejorar en gran medida los resultados humanos añadiendo además la robustez que otorga la unicidad de algunos rasgos biométricos.

En la actualidad, sistemas biométricos como los basados en huella dactilar o iris obtienen unas tasas de reconocimiento formidables siendo la elección preferida en muchos campos de aplicación. Uno de los inconvenientes de estas técnicas es el problema de la adquisición, ya que a menudo resultan demasiado intrusivas para algunas aplicaciones. El reconocimiento facial es una técnica muy estudiada y que, debido a su baja intrusividad y buenos resultados, la convierten en una gran candidata para ser aplicada en entornos donde otras técnicas no son viables. La cara es un rasgo biométrico con gran capacidad discriminativa y con el cual los humanos nos identificamos entre nosotros. Por todo ello, es un rasgo comúnmente aceptado y de fácil adquisición.

Tener la posibilidad de implementar sistemas biométricos en tiempo real abre las puertas a gran variedad de aplicaciones realmente interesantes. Un sistema de reconocimiento que funciona en tiempo real permite aumentar la rapidez y comodidad de uso. Un ejemplo de un sistema en tiempo real sería el reconocimiento de individuos por medio de videocámaras. De esta forma se podrían obtener respuestas inmediatas cuando el individuo aún se encuentra en la zona que son

cruciales si por ejemplo es un sistema que trata de encontrar fugitivos. El reconocimiento facial recoge una serie de características que lo hace idóneo para estas aplicaciones dada su facilidad de adquisición y sus buenos resultados.

En este proyecto se estudia, desarrolla y documenta un sistema de reconocimiento facial en tiempo real completo utilizando diversas técnicas del estado del arte para obtener los mejores resultados. Obtener un sistema completo funcional permite utilizarlo en diversas aplicaciones sin tener que realizar grandes cambios. El creciente rendimiento de los equipos y los últimos algoritmos de reconocimiento, generan la motivación principal para desarrollar un sistema de este tipo.

1.2. Objetivos y enfoque.

El objetivo principal de este proyecto es el estudio, implementación y evaluación de un sistema completo de reconocimiento facial en tiempo real. Para llevarlo a cabo, se han utilizado diversas técnicas del estado del arte de reconocimiento facial y estudiado su aplicación a sistemas de vídeo aprovechando así las ventajas que el vídeo puede ofrecer. La cara es un rasgo muy identificativo que no requiere ni de un despliegue tecnológico complejo ni de una alta participación del individuo para ser adquirido convirtiéndolo en un rasgo perfecto para realizar un sistema en tiempo real a distancia.

Este proyecto se centrará en el estudio de las diferentes técnicas del estado del arte que existen con respecto al reconocimiento facial y tratará de implementarlas para el desarrollo del sistema. El proyecto dará como resultado dos tipos de sistemas; uno, denominado off-line, capaz de trabajar con imágenes de caras y obtener resultados evaluables. El segundo sistema será el sistema en tiempo real, el cual se valdrá de una interfaz para su control. Las fases que compondrán al sistema serán la adquisición de las imágenes mediante una cámara, la detección de caras en las mismas, el preprocesado para extraer las regiones de interés, la extracción de características y por último el reconocimiento de las mismas. Adicionalmente, el sistema también será capaz de almacenar nuevos individuos en la base de datos para después reconocerlos.

Para comprobar la precisión del sistema, se llevarán a cabo una serie de experimentos con bases de datos del estado del arte como BANCA [3] y FOCS [4] y se buscarán las técnicas que mejores resultados ofrezcan. Las técnicas también serán fusionadas con el objetivo de tratar de mejorar dichos resultados. La implementación final del sistema incluirá las técnicas que mejores tasas de reconocimiento obtuvieron.

Una vez el sistema completo ha sido implementado, se desarrollará una interfaz funcional de análisis que permita evaluar el sistema de forma interactiva y educativa. Se realizará un programa por el cual se pueda controlar al sistema de reconocimiento, poniéndolo en modo verificación, identificación o entrenando nuevos individuos para la base de datos interna. Con este programa se tomarán imágenes de diferentes usuarios potenciales para poner a prueba todas sus funcionalidades. Este programa funcionará a través de una cámara encargada de capturar secuencias de vídeo como datos de entrada.

1.3. Metodología y plan de trabajo.

Para el correcto desarrollo y consecución de los objetivos marcados en el presente Proyecto Fin de Carrera, se ha seguido un plan de trabajo organizado en el tiempo como muestra la

Figura 1.1 y detallado a continuación.

- **Estudio del estado del arte.** Todo inicio fundamental de un proyecto pasa por una etapa de formación en la que se obtienen los conocimientos necesarios para su desarrollo. Para este proyecto en concreto, se ha estudiado el estado del arte en reconocimiento facial incluyendo papers y sistemas comerciales. Puesto que el proyecto pone gran énfasis en el desarrollo de un sistema completo, en la etapa de formación también se ha hecho un extensivo estudio sobre los entornos y librerías utilizados en el mismo, OpenCV [5] y Qt [6]. El correcto manejo de los mismos son la base para poder aplicar todos los algoritmos y métodos del sistema.
- **Desarrollo del sistema de reconocimiento.** La etapa central del proyecto es el desarrollo del sistema por el cual el programa va a realizar todas las funciones de reconocimiento. Las diferentes etapas del sistema se van desarrollando con la mayor modularidad posible para que, llegado el momento, se pueda sustituir un algoritmo por otro con sencillez. El desarrollo del sistema se realiza íntegramente en C++ con la incorporación de las librerías de OpenCV que proporcionan funciones de tratamiento de imagen.
- **Ajustes de rendimiento del sistema.** Con el objetivo de comprobar la eficacia del sistema desarrollado, se realizan una serie de experimentos para así obtener resultados sobre la capacidad de reconocimiento en diferentes bases de datos.
- **Desarrollo de la interfaz.** El sistema desarrollado será implementado en un programa que valdrá como estructura externa controlando todos los módulos de trabajo en segundo plano. Esta etapa se separa del desarrollo del sistema, puesto que se realiza en un entorno diferente utilizando la plataforma Qt de desarrollo de programas con interfaces gráficas. Uno de los aspectos clave de esta etapa es la fusión de las librerías de OpenCV con las funciones y lenguaje de Qt para que la transición entre una y otra sea continua y sin obstáculos.
- **Documentación y escritura de la memoria.** En la etapa final del proyecto, se procede a documentar todo el trabajo realizado y a condensarlo en forma de memoria para que este pueda ser entendible y reproducible sin material adicional al citado en la misma.

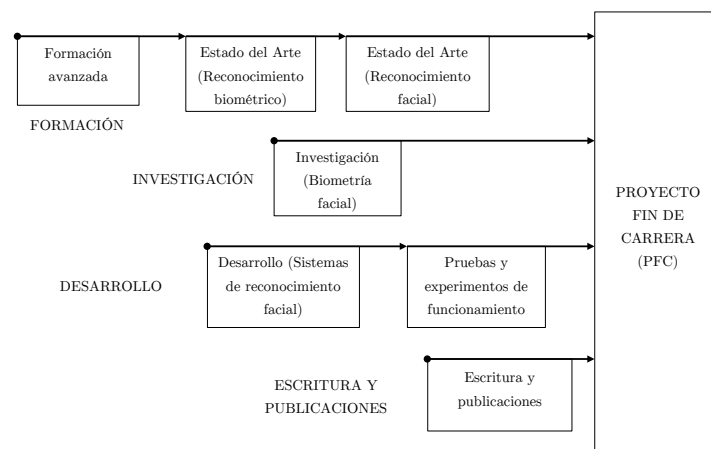


Figura 1.1: Diagrama del plan de trabajo seguido.

1.4. Organización de la memoria.

El presente documento se divide en cinco capítulos:

- **Capítulo 1: Introducción.** En este capítulo se exponen las razones que han motivado a la realización del proyecto así como los objetivos perseguidos para la consecución del mismo. Para lograr satisfacer las motivaciones y objetivos, se expone la metodología y el plan de trabajo que se va a seguir.
- **Capítulo 2: Estado del arte.** Los fundamentos y trabajos relacionados son detallados en este capítulo. Inicialmente se explican los fundamentos del reconocimiento facial añadiendo un trasfondo histórico exponiendo los hitos más relevantes al reconocimiento facial hasta el presente. Se define además, el reconocimiento facial en términos comparativos respecto a otros sistemas de reconocimiento biométrico. Por último y debido al enfoque de desarrollo e implementación del proyecto, se incluye un breve repaso de aplicaciones comerciales desarrolladas y los SDKs más conocidos.
- **Capítulo 3: Sistemas desarrollados.** En este capítulo se exponen los dos sistemas que han sido desarrollados durante la duración del proyecto. En cada sistema se describen las etapas de las consta para su funcionamiento y de las diferentes técnicas implementadas en cada una. Adicionalmente se detalla el equipamiento utilizado para el sistema implementado en el laboratorio.
- **Capítulo 4: Experimentos realizados y resultados.** Los experimentos que se llevan a cabo para medir diferentes características del funcionamiento del sistema se dividen en este capítulo en tres secciones; experimentos relacionados con detección, con reconocimiento y con los tiempos de procesado. Las bases de datos utilizadas para la realización de los dos primeros experimentos se describen junto a los protocolos experimentales.
- **Capítulo 5: Conclusiones y trabajo futuro.** En este capítulo se resume el trabajo realizado en el proyecto y se proponen líneas de trabajo futuras que hagan posible la continuación y mejora del trabajo aquí expuesto.

Para completar el documento, se añade una serie de apéndices con información adicional como un manual de utilización del programa desarrollado, un manual del programador y una guía para configurar los entornos de programación utilizados y poder editar el código.

1.5. Contribuciones.

Las contribuciones de este proyecto se resumen en los siguientes puntos:

- Resumen del estado del arte de SDKs y sistemas comerciales de reconocimiento facial.
- Diseño, desarrollo e implementación de un sistema off-line de reconocimiento facial.
- Diseño, desarrollo e implementación de un sistema de reconocimiento facial en tiempo real junto al equipamiento necesario para su funcionamiento.
- Desarrollo de una interfaz gráfica configurable para controlar el sistema.

- Evaluación de los sistemas obteniendo resultados y conclusiones sobre su rendimiento y comportamiento.
- Implantación de los sistemas desarrollados en el servidor del grupo de investigación donde se desarrolla el proyecto.
- Manual del programador para el desarrollo de los sistemas.
- Manual de configuración e implantación del entorno Qt para utilizar las librerías OpenCV.

2

Estado del arte.

2.1. Introducción al reconocimiento facial en entornos reales

El reconocimiento facial es una técnica que ha tenido un uso muy extendido incluso desde antes de desarrollarse sistemas de reconocimiento automático. El reconocimiento automático de individuos a partir de sus rasgos característicos desde siempre ha suscitado el interés de la comunidad científica puesto que esto permite obtener mayores niveles de eficiencia. La cara es un rasgo muy discriminativo por el cual se pueden identificar individuos a simple vista y que ha sido utilizado como rasgo de identidad en diversos ámbitos. Algunos usos destacables de reconocimiento a partir de la cara son, entre otros, el control de aduanas de los aeropuertos, las entradas de zonas restringidas de ciertas empresas (ver Figura 2.1) o la identificación de sospechosos en la policía.



Figura 2.1: Sistema de reconocimiento facial.

La capacidad de automatizar este proceso consiguiendo resultados incluso mejores que los humanos ha sido el objeto de numerosas investigaciones, obteniéndose así numerosas técnicas de reconocimiento que convierten al reconocimiento facial en un sistema bastante fiable. En la actualidad, uno de los objetivos de estos sistemas es llevar esta fiabilidad obtenida mayoritariamente en entornos controlados, al reto que presentan la variabilidad e instantaneidad de los entornos reales. Con el creciente aumento del rendimiento de la tecnología y la aparición de nuevas técnicas que optimizan el rendimiento así como la carga computacional de los algoritmos de reconocimiento, nace la posibilidad de desarrollar sistemas de reconocimiento que puedan trabajar en tiempo real identificando individuos en cuestión de segundos en entornos no controlados.

El uso de sistemas de reconocimiento facial en tiempo real permite reducir la intrusividad de la obtención de imágenes de individuos así como obtener una respuesta inmediata en un entorno que así lo requiera. La aplicación de estos sistemas en un aeropuerto integrándolos en las cámaras de videovigilancia, permite obtener información sobre posibles sospechosos registrados por la policía en cuanto aparecen dentro de la imagen. La posibilidad de ser verificado en la aduana con una cámara y el pasaporte automáticamente también se hace posible gracias a estos sistemas. El uso en entornos móviles hace posible al dispositivo reconocer al usuario que simplemente lo está mirando y permitir o no el acceso al mismo.

2.2. Historia del reconocimiento facial

Una de las formas más comunes que desde siempre han tenido los humanos para identificarse entre ellos es mediante sus rostros. Al ser un rasgo muy característico y de fácil acceso para la vista humana, somos capaces de discernir entre diferentes personas sólo con la información del mismo. Desde los comienzos de la visión artificial, el reconocimiento facial ha sido estudiado debido a su importancia práctica e interés teórico de científicos cognitivos. A pesar de que otras técnicas biométricas como huella o iris puedan ser más precisas, el reconocimiento facial siempre ha sido muy investigado debido a su naturaleza no invasiva y a que es el método básico de identificación humana. El comienzo de las investigaciones en esta técnica se remonta a los años 60, cuando Woodrow W. Bledsoe y su equipo de investigación [7] desarrollaron los primeros sistemas semi automáticos de reconocimiento basados en el marcado de puntos característicos.

En 1989, T. Kohonen [8] definió la técnica de reconocimiento basada en la caracterización de la cara por la extracción de los autovectores de la matriz de autocorrelación, actualmente conocidos como *eigenfaces*. Esta técnica dio buenos resultados en imágenes de caras alineadas pero no obtuvo los mismos resultados en imágenes donde la situación de la cara no era conocida. Kirby y Sirovich [9] introdujeron un algoritmo para facilitar el cálculo de las *eigenfaces* y demostraron que con menos de 100 era posible caracterizar caras alineadas. En 1991, Turk y Pentland [10] demostraron que el error residual de codificar las *eigenfaces* se podía utilizar para detectar caras en imágenes no controladas y determinar su posición en las mismas. Este método de detección combinado con las técnicas de reconocimiento anteriormente definidas, resultaron en la obtención de un sistema de reconocimiento fiable capaz de trabajar con imágenes en entornos poco controlados. A raíz de estas investigaciones, el interés sobre las técnicas de reconocimiento facial aumento sensiblemente y, desde entonces, numerosas técnicas han sido desarrolladas en este área.

2.3. Fundamentos del reconocimiento facial.

La cara humana, como rasgo característico, nos proporciona gran cantidad de información discriminativa sobre un sujeto permitiéndonos discernir e identificar a simple vista diferentes individuos. La cara alberga, a su vez, un conjunto de rasgos que la dotan de un alto poder discriminativo. Estos rasgos que componen la cara (ejemplo en Figura 2.2), están localizados en posiciones similares a lo largo de la población por lo que un sistema de reconocimiento facial puede beneficiarse de esta característica.



Figura 2.2: La cara como rasgo biométrico.

A continuación se describen los rasgos más significativos que componen el rostro humano.

- **Orejas.** Las orejas están situadas en los laterales de la cara. Habitualmente la variabilidad que presentan entre individuos es eminentemente geométrica, siendo el tamaño la característica que mejor las define. Debido a su localización, las orejas pueden estar ocluidas por el pelo, generando variaciones no deseadas. Es por ello por lo que en muchos sistemas de reconocimiento la región de la cara que se extrae excluye a las orejas para evitar esta variabilidad.
- **Cejas.** Compuestas por vello situado en la parte superior de la cara justo encima de los ojos, ofrecen diferentes características a tomar en cuenta como son el grosor, la forma, el espesor y el color del vello. Su localización puede estar modificada por la expresión aunque por lo general no existe mucha variación del resto de características frente a diferentes gestos.
- **Ojos.** Los ojos, dada su complejidad, son quizá unos de los rasgos más discriminativos de la cara. Situados en la mitad superior de la cara, están compuestos por pestañas, párpados y el globo ocular que a su vez se diferencia en córnea, iris y pupila. Los ojos ofrecen gran variabilidad entre sujetos puesto que su geometría es diferente para cada uno y el iris, un rasgo biométrico por si solo, dota a los ojos de gran información discriminativa. El inconveniente de los ojos es que en ocasiones los párpados ocluyen parcial o totalmente este rasgo, y además son bastante sensibles a cambios de expresión.
- **Nariz.** La nariz está situada aproximadamente en el centro de la cara. Su forma varía en gran medida entre los usuarios y la misma no suele ser afectada en los cambios de expresión. Los dos orificios nasales suelen ser un buen punto característico cuando se miden distancias.

- **Boca.** Por último, la boca situada en la parte inferior de la cara, es otro rasgo característico que facilita información del individuo. Como característica particular, debido a la gran flexibilidad y diversidad de movimientos que puede realizar este rasgo, es posible encontrar gran variabilidad en un mismo sujeto dependiendo de si está sonriendo, si tiene la boca abierta, está sacando la lengua, etc. Los labios son el componente que siempre está visible y que suelen definir el aspecto de la boca.

Adicionalmente a estos rasgos, también cabe destacar que la forma de la cara también es una característica discriminativa, así como otras zonas de la misma como pueden ser los pómulos, la frente o la barbilla. La cara es un rasgo con una componente simétrica bastante elevada, algo de lo que las tareas de localización o extracción de distancias se pueden beneficiar.

Existen, sin embargo, ciertas características de la cara que pueden introducir mayor variabilidad en el mismo individuo. En este sentido, el elemento más destacado es el pelo. El pelo puede contribuir a la oclusión de los rasgos (ya sea cabello o barba) y a cambiar el aspecto de una persona. Otros elementos artificiales comunes que suelen contribuir a la pérdida de fiabilidad en el sistema son las gorras, bufandas y gafas, siendo las gafas de sol las que mayor oclusión generan.

2.4. Ventajas e inconvenientes del reconocimiento facial.

El reconocimiento facial presenta ciertas características favorables que la convierten en una técnica más viable para ser utilizada en ciertos ámbitos respecto a otras técnicas biométricas [11]. Los aspectos más determinantes se definen a continuación.

- **Baja intrusividad.** La cara, al ser un rasgo humano muy visible, facilita la tarea de su obtención lo que conlleva a que la misma sea poco intrusiva para los individuos. La cooperación que conlleva esta obtención es también mínima puesto que con una cámara sea capaz de capturar la cara de una persona con una calidad aceptable es suficiente. Esto permite diseñar sistemas de reconocimiento sin que los individuos se percaten de que están siendo identificados.
- **Gran poder discriminante.** Aunque la cara no sea uno de los rasgos con mayor poder discriminativo, sí que es un rasgo con una cantidad suficiente de información y variabilidad intra clase para que tenga un alto poder discriminante. Una alta cantidad de sujetos en una base de datos no llega a deteriorar en gran medida la eficacia del sistema obteniéndose también en estos casos buenas tasas de reconocimiento.
- **Disponibilidad extendida.** El ser humano ha utilizado como método de reconocimiento básico la cara de las personas en gran variedad de ámbitos. Esto hace que haya una gran disponibilidad de bases de datos y que este rasgo haya sido obtenido de la mayor parte de la población con anterioridad. Al haber sido este rasgo muy utilizado, se han llevado a cabo multitud de investigaciones referentes al reconocimiento facial que han resultado en diferentes técnicas y algoritmos de los cuales hoy en día nos podemos beneficiar.

Al igual que las ventajas aquí descritas, el reconocimiento facial, como cualquier otra técnica biométrica, presenta ciertos inconvenientes que deben ser mencionados. Uno de los grandes inconvenientes que presenta esta técnica, es el hecho de que existe variabilidad intra-clase. Esta

variabilidad viene dada por la plasticidad del rostro humano respecto a los gestos y a que es un rasgo que no presenta invariabilidad temporal [12], teniendo varios rostros de un sujeto a lo largo de los años diferentes a simple vista. Cuando el entorno no está controlado, los cambios en la pose, oclusiones (gafas, gorro etc.) y baja calidad de las imágenes tomadas, hacen que el sistema pierda fiabilidad presentando inconvenientes a la hora de reconocer personas en un entorno real. Por último, existen ciertas situaciones particulares en los que el reconocimiento facial no es viable como en el caso de que dos personas sean gemelas o, debido al fácil acceso del rasgo, un impostor podría utilizar imágenes de rostros de otras personas sin mucha dificultad, aunque en este sentido, en los últimos años se está prestando mucho interés estudiando y generando soluciones.

Cabe destacar que respecto a varios de los inconvenientes anteriormente descritos, año tras año se están desarrollando nuevas técnicas e investigando formas de solventarlos que hacen disminuir el impacto que tienen sobre los sistemas de reconocimiento facial. En el actual estado del arte se están consiguiendo tasas de reconocimiento razonablemente altas para imágenes tomadas en entornos no controlados [13].

2.5. Aplicaciones de los sistemas de reconocimiento facial.

- **Control de acceso.** Una de las aplicaciones de los sistemas de reconocimiento facial más extendidas son el uso de los mismos en lugares donde se requiere la verificación de identidad. Un ejemplo es la verificación de la identidad de una persona en la aduana de un aeropuerto o, como muestra la Figura 2.3 el control a la entrada de un estadio. La información contenida en el documento de identidad de la persona puede ser contrastada automáticamente con una imagen de su cara. Existen a su vez empresas con áreas de acceso restringido en las que los sistemas de reconocimiento facial también son útiles ya que en este caso existiría una base de datos previa con las fotos de las personas autorizadas a las cuales el sistema solo garantiza el acceso por medio de la obtención de la imagen de su cara en el lugar de acceso.

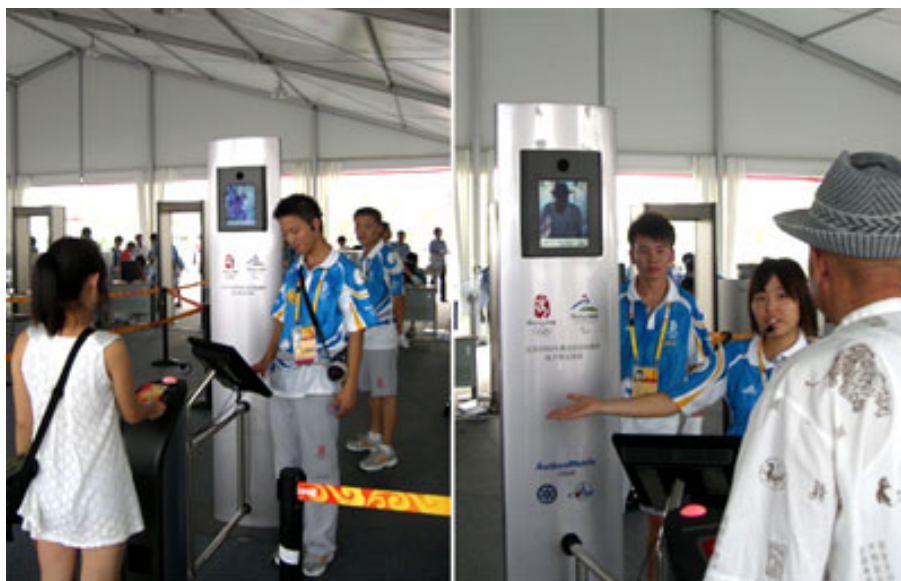


Figura 2.3: Sistema de reconocimiento facial en los juegos olímpicos de Pekín.

- **Sistemas de seguridad.** Estos sistemas también son utilizados en circuitos cerrados de televisión para añadir una capa más de seguridad al sistema de videovigilancia. Las cámaras son capaces de detectar las caras de los individuos que aparecen en la imagen y realizar tareas de reconocimiento en tiempo real contra una base de datos de sospechosos y alertar en caso de obtener alguna coincidencia. La utilización de estos sistemas está facilitada por la gran extensión de los sistemas de vigilancia a infinidad de áreas.
- **Biometría forense.** Las grandes bases de datos de individuos fichados por la policía hacen que los sistemas de reconocimiento facial en este entorno sean de gran importancia (ejemplo en la Figura 2.4). Sistemas capaces de verificar la identidad de un sospechoso que acaba de ser detenido o sistemas capaces de comprobar si la persona tiene antecedentes se convierten en básicos y necesarios para este tipo de aplicaciones. Aparte de trabajar con sus datos, estos sistemas deben de ser capaces de ser compatibles con cualquier información que sea enviada a la policía para realizar las tareas de reconocimiento. Ardua tarea debido a la cantidad de escenarios reales con gran variabilidad que la mayoría de veces se han de analizar.



Figura 2.4: Identificación de individuos durante los disturbios de Londres.

- **Aplicaciones móviles y redes sociales.** Con la llegada de los *smartphones* al mercado, los teléfonos móviles son cada día capaces de realizar una mayor variedad de tareas debido al desarrollo de la tecnología. Los teléfonos con cámara integrada son capaces de beneficiarse de sistemas de reconocimiento facial para permitir el acceso al dispositivo por parte del dueño o para, por ejemplo, detectar las caras en una foto y reconocer si se trata de gente guardada en la lista de contactos para utilizar esa información como clasificador. Muchos programas de fotografía también llevan integrados sistemas de reconocimiento facial para automatizar las tareas de compartir fotos en redes sociales.

2.6. Etapas de un sistema de reconocimiento facial.

Tal y como muestra la Figura 2.5, los sistemas de reconocimiento facial se dividen en cinco etapas principales. La primera es la adquisición de los datos mediante cámaras, webcams, video cámaras o cualquier dispositivo que proporcione una fotografía que contenga una cara a analizar. La segunda etapa comprende la detección y localización de la cara para su posterior preprocesado en la tercera etapa. Esta etapa de preprocesado se lleva a cabo para preparar los datos mediante



Figura 2.5: Diagrama general de un sistema de reconocimiento facial.

la normalización, alineación y escalado de la imagen. Una vez que los datos están preparados, se realiza la cuarta etapa de extracción de características para obtener la información relevante de la imagen. Por último, la etapa de reconocimiento donde se aplican los algoritmos de reconocimiento y se extrae una decisión respecto a la base de datos y la imagen de entrada.

A continuación se exponen los diferentes métodos que existen en el estado del arte actual para implementar dichas etapas.

2.6.1. Adquisición de la imagen.

La adquisición de las imágenes de entrada se realiza mediante cualquier dispositivo capaz de tomar imágenes como, por ejemplo, una cámara fotográfica, una cámara de videovigilancia o la cámara integrada de un dispositivo móvil. Como con una imagen basta para comenzar el proceso, el tiempo de adquisición es muy bajo y no requiere de supervisión siempre y cuando las imágenes tomadas cumplan con un mínimo de calidad.

2.6.2. Detección.

La etapa de detección en los sistemas de reconocimiento facial es crítica puesto que el resto de etapas se verán afectadas si no se ha realizado una detección y localización correctas. La detección está formada por dos partes:

- **Detección de la región de la cara.** La detección de las regiones de interés en una imagen se realiza mediante los denominados *Haar-like features* mostrados en la Figura 2.6, adaptados por Viola y Jones a partir del uso de *Haar wavelets* [14]. Este sistema considera regiones rectangulares en una ventana de detección, suma las intensidades de los píxeles en cada región y calcula la diferencia entre estas sumas. La diferencia es usada para clasificar subsecciones de la imagen.

La ventaja de este sistema desarrollado por Viola y Jones es la rapidez de cálculo, haciendo posible la detección de objetos en tiempo real. Este proceso requiere de un entrenamiento previo con una gran cantidad de imágenes positivas (caras) e imágenes negativas (imágenes sin caras) para generar lo que se denomina un clasificador en cascada. Con un entrenamiento correcto, se hace posible un detector de caras robusto como se muestra en la Figura 2.7.

- **Detección de la posición de los ojos.** Para realizar una correcta alineación de la imagen en el preprocesado, es necesario determinar las coordenadas de los ojos en la misma. Para detectar los ojos existen varios métodos. El método más directo es utilizar los clasificadores Haar usados en la detección de caras pero esta vez entrenados con imágenes de ojos [15]. El segundo método consiste en acotar la región en la que sabemos que van a estar los ojos por geometría facial y obtener la gráfica que representa la suma por columnas de los

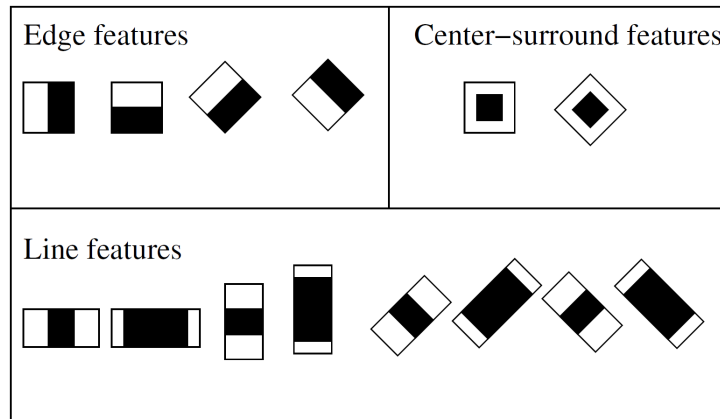


Figura 2.6: Características tipo Haar.

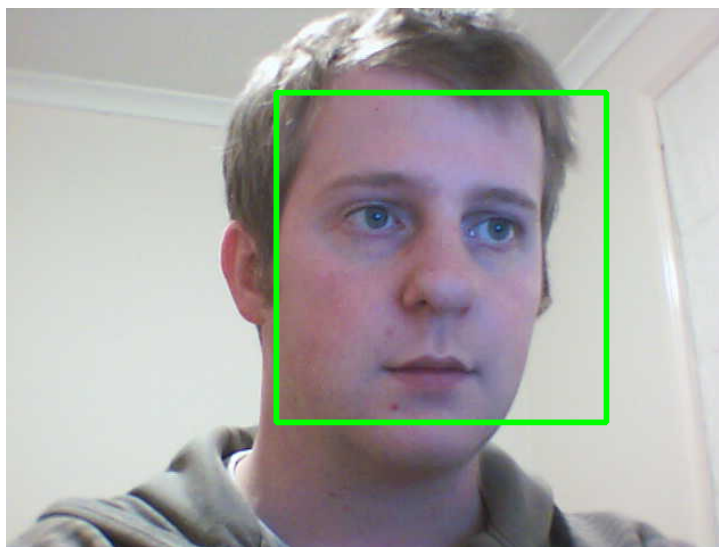


Figura 2.7: Detección de la región de la cara.

valores de la imagen binarizada. De esta forma se pueden buscar los picos de la imagen que corresponderán a los ojos.

Por último, con métodos basados en momentos invariantes [16] también se obtienen buenos resultados. Para poder detectar los ojos de esta forma, la imagen se binariza y se etiquetan las regiones inconexas de la misma. Posteriormente, se utilizan los momentos invariantes para conseguir obtener las dos regiones más parecidas.

2.6.3. Preprocesado y normalización.

La etapa de preprocesado [17] se lleva a cabo a partir de la información obtenida en la detección. Esta etapa realiza una serie de transformaciones geométricas sobre la imagen dejándola preparada para la correcta extracción de características. En el preprocesado se utilizan cuatro fases para normalizar y alinear la imagen tal y como muestra la Figura 2.8.

- **Rotación.** Una de las utilidades de calcular las coordenadas de los ojos, radica en poder determinar el ángulo de giro de una cara en una imagen y compensarlo. Al tener caras sin

giro, el proceso de reconocimiento dará mejores resultados.

- **Escalado.** Para conseguir que todas las imágenes tengan el mismo tamaño, se utiliza la distancia entre los centros de los ojos para conseguir un ratio por el cual la imagen debe ser aumentada o reducida. Esto es necesario puesto que muchas técnicas de reconocimiento requieren que todos los datos de entrada tengan el mismo tamaño (En nuestro caso la matriz de píxeles).
- **Recorte.** Una vez la imagen ha sido rotada y escalada, se procede al recorte de la misma para obtener sólo la región de interés. Para definir la región se utiliza la coordenada del ojo derecho. Existen varios tamaños de imagen estandarizados por los cuales se puede extraer la región de interés relativa a las necesidades del sistema. Los formatos de imagen están recogidos por el estándar ISO/IEC 19794-5 [18] que define un área similar a una foto de carnet, en base a dicho estándar se establece una región que comprende exclusivamente el área de la cara.
- **Normalización de histograma.** Las imágenes pueden presentar variabilidad en la luminosidad y en el contraste lo que produce que imágenes similares sean muy diferentes respecto al valor de intensidad de sus píxeles. Mediante la normalización de su histograma, se pretende que las imágenes que tienen la mayor parte de sus valores de intensidad concentrados en una zona reducida del histograma, pasen a extenderse por todo el rango de valores del histograma. Esto resulta en imágenes con mayor contraste y con menor variabilidad lumínica entre ellas.

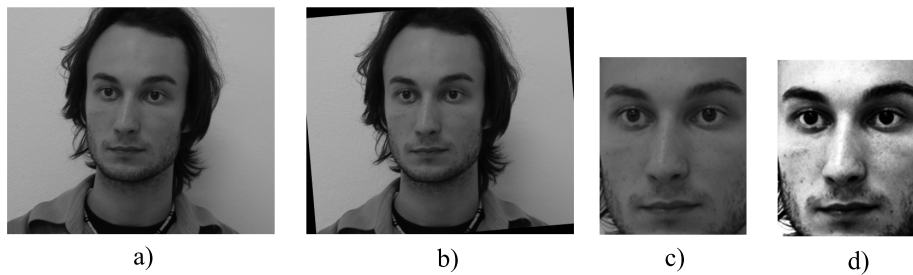


Figura 2.8: Preprocesamiento de una imagen. a) Imagen de entrada. b) Rotación a partir de las coordenadas de los ojos. c) Recorte y escalado en base al estándar ISO/IEC 19794-5. d) Ecualización del histograma.

2.6.4. Extracción de características.

La extracción de características se emplea para obtener la información que resulta relevante de cara a realizar una comparación. Como se puede ver en la Figura 2.9, los métodos de extracción de características no dependientes de pose se dividen en tres grandes grupos, métodos basados en apariencia, métodos basados en puntos característicos de la cara y métodos híbridos. En este proyecto no se estudiarán los algoritmos independientes de pose puesto que la mayoría de ellos requieren equipamiento especializado para la captura de imágenes o capturas muy específicas y controladas.

1. **Métodos holísticos o basados en apariencia.** Estos métodos utilizan toda la región de la cara como entrada del sistema de reconocimiento. La imagen de la cara es transfor-

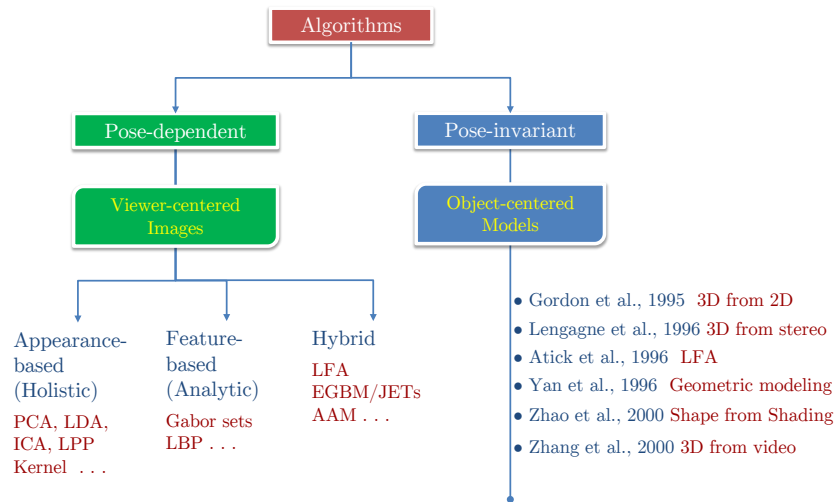


Figura 2.9: Taxonomía del reconocimiento facial.

mada a un espacio en el cual se aplican técnicas estadísticas. Al utilizar toda la cara como único rasgo, estos métodos suelen presentar limitaciones dadas por cambios de expresión, pose o iluminación. A continuación se describen brevemente algunas de las técnicas más usadas para los métodos basados en apariencia.

- **Principal Component Analysis.**[19] Una técnica muy popular es el *Principal Component Analysis* (PCA) que transforma la imagen a un subespacio (ejemplo en Figura 2.10) por el cual es posible obtener vectores de características de menor dimensionalidad sin una pérdida de información discriminativa importante.



Figura 2.10: Eigenfaces obtenidas a partir de la proyección de las imágenes en el espacio PCA.

- **Linear Discriminant Analysis.**[20] Derivada del PCA, se implementó una técnica que aúna PCA con *Linear Discriminant Analysis* (LDA) denominada *fisherfaces*, con el cual se consigue mayor robustez frente a cambios de iluminación.

- **Frequency Domain Analysis.**[21] Por último, las técnicas de análisis en el dominio de la frecuencia ofrecen una representación de la imagen en función de las componentes de baja frecuencia que presentan alta energía. Tales técnicas como DFT, DCT o DWT son independientes de los datos por lo que no requieren de un entrenamiento. Adicionalmente, existen algoritmos optimizados que facilitan la implementación y reducen el coste computacional.

2. **Métodos locales o basados en puntos característicos de la cara.** Éstos métodos se basan en extraer los rasgos que componen la cara como la nariz, los ojos, o la boca para clasificar sus características geométricas y/o de apariencia por separado en el sistema.

- **Análisis de distancias a puntos característicos.**[22] Uno de los primeros sistemas de reconocimiento se basa en la técnica a partir de puntos geométricos de la cara. A partir de la detección de diferentes puntos característicos (por ejemplo, como se muestra en la Figura 2.11 localización de los lagrimales, agujeros de la nariz, comisura de los labios, etc.) se crean vectores que contienen datos de distancias entre los mismos. Cuantos más puntos característicos son detectados, mayor número de distancias podrán ser calculadas obteniendo así mejores resultados en el reconocimiento.

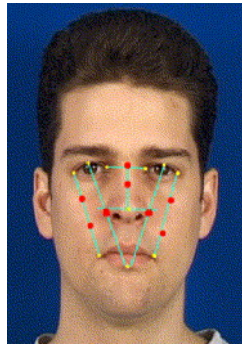


Figura 2.11: Vectores de distancias obtenidos a partir de puntos característicos.

- **Local Binary Patterns.**[23][24] El algoritmo de LBP es sencillo pero dota a la información de gran robustez frente a cambios de iluminación (ver Figura 2.12). Se basa en ir tomando vecindarios respecto a un píxel central, el cual establece un valor de umbral. El vecindario es binarizado dependiendo de si el valor es mayor o menor que el umbral y cada valor así hallado es concatenado para formar un sólo número binario que más tarde se pasa a decimal siendo este número el nuevo valor del píxel. La imagen se divide en regiones donde se aplica LBP y se obtiene su histograma. Estos histogramas son posteriormente concatenados para obtener una representación de la cara.

- **Elastic Bunch Graph Matching.**[25] EBGM es una técnica que aprovecha la estructura topológica similar de las caras. Básicamente, las caras son representadas como grafos con los nodos situados en los puntos característicos. Los vértices se etiquetan con la distancia de cada nodo conteniendo un set de 40 coeficientes complejos de *wavelets* de Gabor en diferentes escalas y orientaciones. EBGM usa estos grafos para representar una cara humana y codifica la apariencia local usando *wavelet jets*.

- **Hidden Markov Models.**[26] Los HMMs también han sido usados con éxito para el reconocimiento facial. Los HMM presentan robustez frente a cambios de iluminación, expresión y orientación, otorgando así una ventaja frente a los métodos holísticos. Las técnicas basadas en HMM utilizan regiones horizontales de píxeles que albergan a la frente,

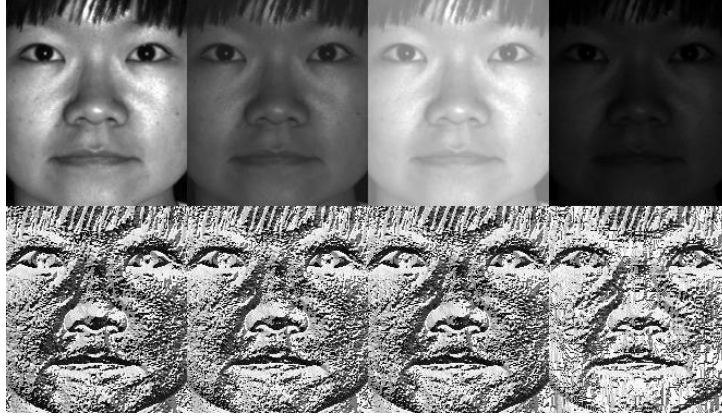


Figura 2.12: LBP sobre imágenes con diferente intensidad.

ojos, nariz, boca y barbilla sin obtener la posición exacta de cada rasgo. Cada una de estas regiones es asignada a un estado del HMM para el reconocimiento.

3. **Métodos híbridos.** Estos métodos realizan una fusión de los dos métodos antes descritos para obtener una mejora en los resultados. Un ejemplo de este método es el resultado de realizar PCA tanto a la cara global como a cada rasgo por separado, obteniéndose *eigenfaces* y *eigenfeatures* que describen la cara [4].

Este proyecto se centra en el uso de métodos tanto basados en apariencia como locales para la realización de un sistema de reconocimiento completo.

2.6.5. Comparación y Reconocimiento.

Para obtener una puntuación por la cual tomar una decisión, existen varios métodos de comparación. En este apartado se describen algunos de los más utilizados en el estado del arte. Los métodos se pueden dividir en medidas de similitud y clasificadores[27].

1. Medidas de similitud o distancia.

- **Distancia Euclídea** La distancia euclídea es una de las medidas más básicas para calcular distancias. Esta distancia se define como la distancia directa entre dos puntos en un plano. El ejemplo más claro es la distancia entre dos puntos en un plano de dos dimensiones de coordenadas x e y . Si tuviéramos dos puntos P_1 y P_2 con coordenadas x_1, y_1 y x_2, y_2 respectivamente, el cálculo de la distancia euclídea entre los mismos sería:

$$d_E(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- **Chi-Square (χ^2)** La distancia Chi-Square toma este nombre ya que la fórmula que la calcula es prácticamente idéntica a la prueba de bondad de ajuste χ^2 utilizada para comparar distribuciones de probabilidad discretas. En el caso de reconocimiento biométrico, χ^2 se utiliza para medir la distancia entre dos vectores de características creados a partir de histogramas. El cálculo de la distancia para el caso de dos histogramas S y M es:

$$\chi^2(S, M) = \sum_i \frac{(S_i + M_i)^2}{S_i + M_i}$$

2. Clasificadores.

- **K-Nearest Neighbours.**[28] El método k-NN es una técnica no paramétrica de clasificación de objetos basado en las muestras de entrenamiento más cercanas del espacio de características, El algoritmo se basa en encontrar los k vecinos más cercanos al objeto para, en función de la cantidad de los mismos, clasificarlo en el conjunto que tenga un mayor número de muestras cercanas.

- **Support Vector Machines.**[29] Los SVMs son modelos de aprendizaje usados para clasificación y regresión. El objetivo de este método es el de representar en un espacio una serie de clases y tratar de encontrar un hiperplano que las divida de forma que cree zonas en las cuales, cuando entra un dato, sea clasificado por dichas zonas. El espacio donde se representan las clases tendrá un número de dimensiones igual al número de clases, por lo que un ejemplo sencillo de representar es el caso de clasificación de dos clases.

Tal y como muestra la Figura 2.13, las clases se distribuyen por el plano en dos dimensiones y los SVM tratan de buscar un hiperplano (en este caso una recta) que divida ambas clases de una forma óptima, es decir, que la distancia entre ambas sea máxima. El cálculo de este hiperplano depende en gran medida de la distribución de las clases, puesto que el ejemplo expuesto es un caso ideal en el que hay una gran separación entre clases, pero en los casos reales las clases suelen tener muestras mezcladas haciendo más difícil la clasificación. Es por ello que, dependiendo de la técnica utilizada para extraer características, los SVMs funcionarán mejor, o en ciertos casos no serán viables.

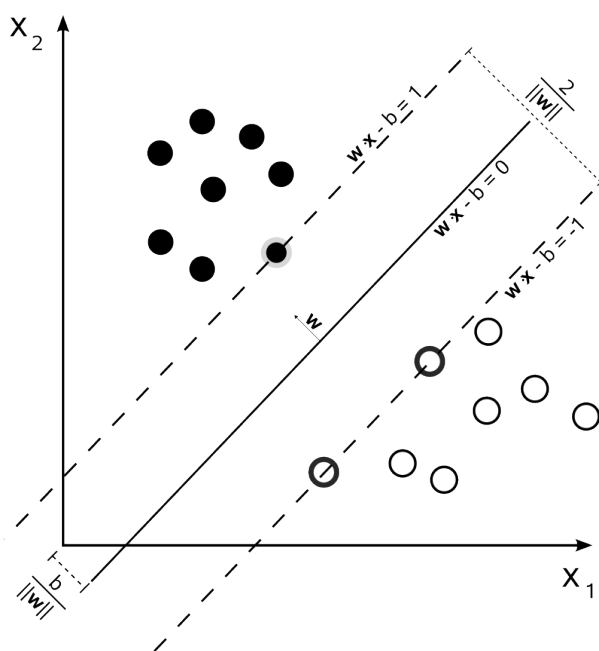


Figura 2.13: Cálculo del plano que mejor divide las dos clases en un problema de dos dimensiones.

- **Gaussian Mixture Models.** Los GMMs son funciones de densidad de probabilidad paramétricas representadas como una suma de componentes gaussianas. En clasificación biométrica se suelen utilizar en conjunto con EM (Expectation Maximization) para estimar sus parámetros. Cabe destacar que OpenCV no tiene aún una implementación de esta técnica.

2.7. Sistemas comerciales y SDKs.

En la actualidad gran cantidad de sistemas de reconocimiento facial en tiempo real están en uso para muy diversas aplicaciones, ya sean comerciales, públicas, de uso general o privadas. Esta sección trata de hacer un repaso de los sistemas en uso más destacados en este área, así como de los SDKs más usados para el desarrollo de los sistemas.

2.7.1. Sistemas comerciales en la actualidad.

Muchos sistemas de reconocimiento facial son desarrollados en la actualidad. En esta sección se detallan algunos de los sistemas más conocidos y las respectivas empresas que los desarrollan (Figura 2.14).



Figura 2.14: Empresas desarrolladoras de sistemas de reconocimiento facial.

Next Generation Identification Programme.

Este proyecto que lanzó el FBI en 2010 pretende instaurar un sistema de reconocimiento facial en cámaras de videovigilancia a lo largo de todo Estados Unidos (Figura 2.15). Informes de 2012 revelan que el proyecto ya ha sido desplegado en un 60 % y sigue en marcha. El principal objetivo del uso de estos sistemas es la capacidad de detectar e identificar fugitivos, personas perdidas o sujetos de interés. Adicionalmente el sistema está preparado para realizar tareas de vigilancia automática en congregaciones del tipo *Occupy Wall Street*. El sistema es capaz de realizar tareas de reconocimiento tanto con bases de datos privadas del FBI (Registros del *National Criminal Information Center*) como con recursos públicos como pudiera ser la red social *Facebook*. Los algoritmos de reconocimiento utilizados para la realización de estos sistemas no se han hecho públicos pero sí que se han facilitado datos que muestran una tasa de reconocimiento del 92 % en bases de datos con 1,6 millones de imágenes.



Figura 2.15: Next Identification Programme.

Reconocimiento Facial en Picasa.

El programa de Google de edición y organización de fotos Picasa, añadió en su versión 3.5 a finales de 2009 la capacidad de reconocer caras recorriendo todas las fotos guardadas en el ordenador donde estuviera instalado. Si el usuario tiene una cuenta de Google con contactos vinculada al programa, el sistema es capaz de obtener los datos de los contactos y etiquetar las fotos en las que logra reconocer sus caras. De esta forma, es posible realizar una organización o filtrado de las fotos en función de las personas que aparecen en las mismas. Cuando el sistema no es capaz de encontrar una coincidencia para una cara detectada, se pide al usuario si desea guardar esa persona como un contacto nuevo y entonces se añadirá a la base de datos para el etiquetado de futuras coincidencias. Las técnicas utilizadas por el programa no son públicas pero debido a que han sido desarrolladas por Neven Vision, un estudio de las patentes que tienen vigentes revelan que están basadas en EBGM.

SmartGate.

A finales de 2007, el servicio de aduanas Australiano instauró el sistema de reconocimiento facial automático SmartGate. El sistema pretende sustituir al control de aduanas humano situado en los aeropuertos que se encarga de verificar la identidad de un individuo con su pasaporte como se puede ver en la Figura 2.16. El funcionamiento de SmartGate es el siguiente; el usuario entra en la cabina de reconocimiento e introduce su pasaporte electrónico donde su foto será obtenida. Posteriormente el sistema toma imágenes del individuo a partir de tres cámaras situadas en diferentes posiciones. El sistema realiza una tarea de verificación con las fotos obtenidas. Si la verificación es positiva, el sistema deja continuar al usuario. Si por el contrario, el sistema no es capaz de verificar la identidad del individuo, un oficial de aduanas se encarga de la situación. Con este sistema se consigue aligerar el proceso de verificación de identidad en las horas punta aunque por el momento sólo está permitido para pasajeros con nacionalidad Australiana.

MorphoFace Investigate.

La empresa de biometría Safran Morpho ofrece gran variedad de productos lanzados al mercado centrados en el reconocimiento biométrico basado en huella dactilar. Esta empresa crea



Figura 2.16: SmartGate implantado en aeropuertos australianos.

aplicaciones, SDKs y sensores para un amplio rango de campos como son controles de aduanas, identificación en empresas o bases de datos policiales. Entre estos productos, también ofrecen un sistema de reconocimiento facial enfocado a la identificación de criminales conocido como MorphoFace Investigate (ver Figura 2.17). Esta aplicación sirve de ayuda a los departamentos de la policía para gestionar las bases de datos de sospechosos con funcionalidades como el poder etiquetar delitos a individuos y así poder optimizar los tiempos de búsqueda. El sistema de reconocimiento permite incorporar nuevas imágenes a la base de datos de forma rápida y eficaz. Cabe destacar que existe una gran sección de la empresa llamado MorphoTrust USA que trabaja conjuntamente con el gobierno de los Estados Unidos desarrollando todo tipo de sistemas biométricos de seguridad para las agencias federales y el ejército.



Figura 2.17: MorphoFace Investigate.

Portátiles Toshiba.

Los últimos modelos de portátiles Toshiba llevan incorporado un sistema de reconocimiento facial para realizar diversas tareas que requieren autenticación como iniciar sesión (Figura 2.18), otorgar permisos de administrador o como sustituto de clave numérica en algunos programas. Sistemas de comportamiento similar pero haciendo uso de la huella dactilar ya habían sido implantados con éxito anteriormente. En este caso, los portátiles con sistema de reconocimiento facial no requieren de un sensor adicional como ocurre con los de huella, ya que se valen de la cámara integrada para realizar estas funciones.



Figura 2.18: Toshiba Face Recognition.

Face Unlock.

De forma similar que con el software de Picasa, en 2011 Google compró a la compañía especializada en reconocimiento facial PittPatt. El objetivo de Google era llevar los sistemas de reconocimiento facial a la plataforma Android para que pudieran ser utilizados por dispositivos móviles. Valiéndose del SDK desarrollado por PittPatt, Android lanzó en 2012 una nueva funcionalidad del sistema operativo, *Face unlock*. Con *Face unlock*, un sistema completo de reconocimiento facial en tiempo real se hizo posible en dispositivos con capacidades limitadas de procesamiento. Esta funcionalidad permite a los usuarios de móviles con Android cambiar el desbloqueo mediante clave por el desbloqueo mediante el reconocimiento de la cara. Para realizar esto, el sistema previamente lleva a cabo una fase de entrenamiento por la cual almacena imágenes de la cara del usuario haciendo uso de la cámara frontal como se muestra en la Figura 2.19.

Una vez se ha guardado correctamente al usuario, el dispositivo realizará tareas de detección y reconocimiento cuando la pantalla está bloqueada y un usuario la enciende. Si el reconocimiento es positivo, el dispositivo permitirá el acceso al usuario. Con este reconocimiento de usuario también se pretende cargar configuraciones y preferencias personalizadas basadas en el uso.

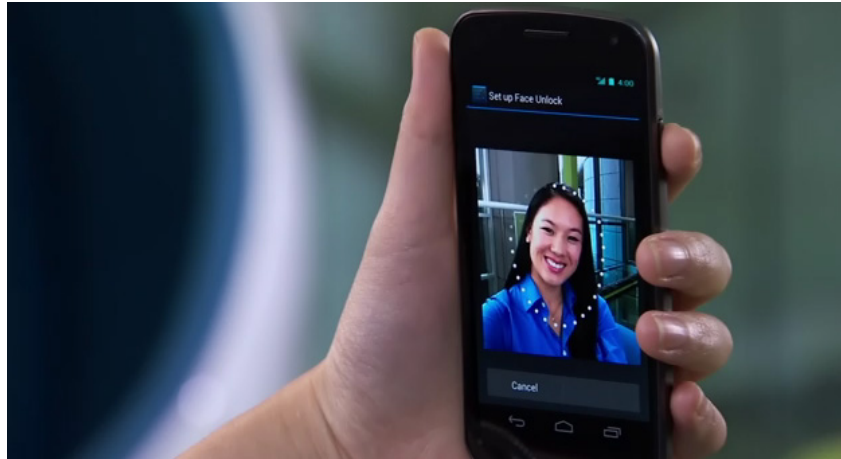


Figura 2.19: Entrenamiento de Face Unlock.

Google Glass.

Un proyecto que está todavía en fase de pruebas, pero que ha despertado el interés de mucha gente son las Google Glass, unas gafas capaces de proyectar información en el cristal y grabar vídeo gracias a su cámara incorporada (Figura 2.20). Este prototipo hace uso de técnicas de realidad aumentada para plasmar la información, y se comunica con el dispositivo móvil del usuario para realizar tareas complejas como búsquedas en internet u obtención de contactos para realizar llamadas sin tener que manejar el dispositivo. Debido a sus cualidades, las gafas están ganando interés en la comunidad científica por todas las aplicaciones que se podrían desarrollar en torno a las mismas, entre ellas sistemas de reconocimiento facial en tiempo real. El reto que se plantea para que esto sea posible es la necesidad de utilizar bases de datos, en ocasiones muy extensas con la carga computacional y de espacio que conllevan. Existen también preocupaciones al respecto puesto que la privacidad se ve comprometida con la aparición de dispositivos de este tipo.



Figura 2.20: Google Glass.

2.7.2. SDKs de reconocimiento facial.

FaceVACS-SDK.

Este SDK desarrollado por Cognitec, permite a los usuarios implementar un sistema de reconocimiento facial en sus aplicaciones. El software trae soporte para múltiples algoritmos relacionados con métodos basados en apariencia. FaceVACS-SDK cubre las funciones de entrenamiento, verificación e identificación. El API es compatible con los lenguajes de programación C++, C, Java y entornos .NET. Asimismo, Cognitec provee a los usuarios del SDK con FaceVACS-Engine, especialmente diseñado para aplicaciones móviles y sistemas embebidos con limitaciones de rendimiento. Entre otros, el SDK contiene numerosas técnicas de detección como detección de ojos, detección de gafas, detección de ojos o boca cerrados, detección de mirada, estimación de género o estimación de edad.

- Disponible en: <http://www.cognitec-systems.de/FaceVACS-SDK.19.0.html>

Perceptual Computing SDK.

En 2012 Intel desarrolló un SDK para imagen artificial que, entre otras librerías, contiene una de reconocimiento facial. Este SDK, escrito en C, contiene una serie de funciones para implementar un sistema completo. El sistema extrae puntos característicos de la cara (Figura 2.21) como son los ojos, la nariz o la boca para realizar la tarea de reconocimiento.

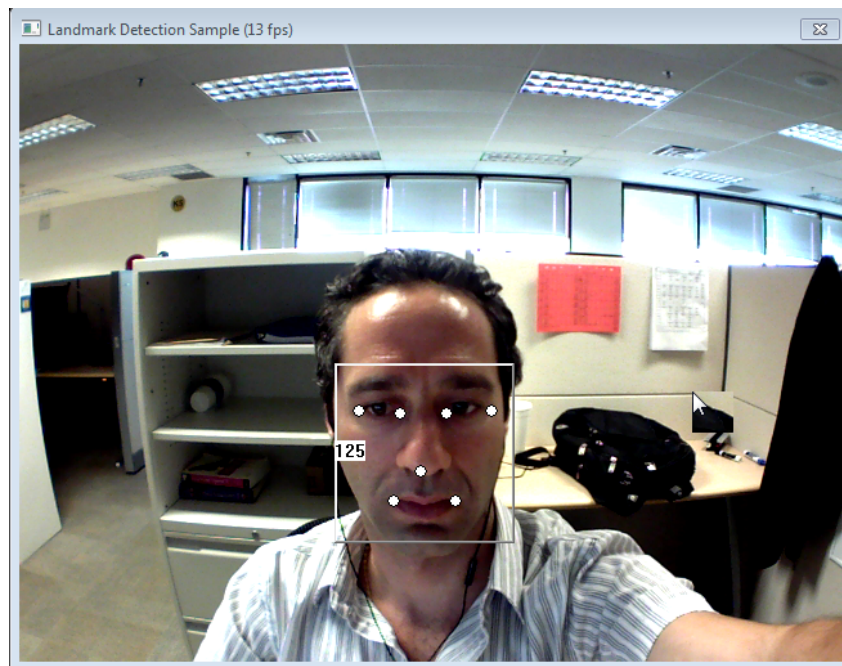


Figura 2.21: Extracción de puntos característicos con Perceptual Computing SDK.

El software es configurable pudiendo determinar cuántos puntos característicos se van a detectar o si la entrada de los datos son imágenes o secuencias de vídeo. Además de la función principal de reconocimiento, el SDK también contiene funciones para determinar edad, género y expresión.

- Disponible en: <http://software.intel.com/en-us/vcsourc/tools/perceptual-computing-sdk>

Verilook SDK.

La empresa Neurotechnology es la creadora del SDK de reconocimiento facial Verilook. Este software cuenta con tres versiones; la versión de escritorio, la versión para móviles y la optimizada para videovigilancia. La versión de videovigilancia ofrece funcionalidades como la posibilidad de tener todos los dispositivos conectados a través de un servidor que contiene la base de datos compartida. El reconocimiento en tiempo real del SDK es compatible con múltiples caras simultáneas en la misma imagen. El entrenamiento del sistema (ver Figura 2.22) se realiza aparte del sistema de reconocimiento.

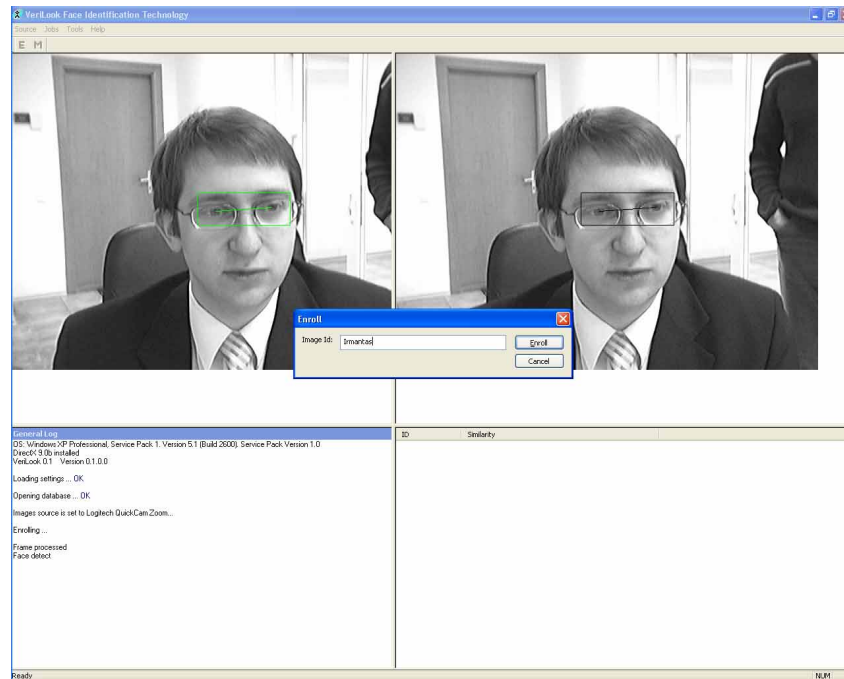


Figura 2.22: Entrenamiento con verilook SDK.

- Disponible en: <http://www.neurotechnology.com/verilook.html>?

FaceIt.

Este SDK desarrollado por Bayometric Inc. contiene todas las funciones necesarias para realizar tareas de detección y reconocimiento en un sistema de reconocimiento facial centrado en equipos de sobremesa y cámaras de videovigilancia. *FaceIt* contiene módulos para detección de cara, creación de plantillas y análisis de calidad todos ellos empaquetados en un interfaz de programación en C. Este SDK es utilizado por la empresa MorphoTrust USA, encargada de varios servicios destinados a la seguridad federal en Estados Unidos.

- Disponible en: <http://www.bayometric.com/products/Face-Recognition-SDK.htm>

Face Forensics.

Face Forensics es un SDK creado por 360 Biometrics que permite implementar sistemas de reconocimiento conectados entre sí a una misma base de datos por medio de conexión a un

servidor. Es compatible con bases de datos en formato SQL, Oracle, D2b o MSAccess y trae una base de datos de entrenamiento por defecto. Su característica de conectividad permite a los diferentes sistemas que implementan este software, actualizar las bases de datos con nuevos entrenamientos de forma simultánea. Además es capaz de enviar automáticamente resultados a cualquier cuenta asociada con el mismo. Como opción adicional, el software introduce la funcionalidad de reconocimiento facial de sólo una zona específica para casos de reconocimiento de víctimas de accidentes.

- Disponible en: <http://360biometrics.com/face-recognition.php>

FaceSDK.

La empresa de biometría Luxand tiene disponible un SDK de reconocimiento facial para crear aplicaciones multiplataforma. El software es compatible con Windows, Mac OS y Linux. Este SDK incluye integración con un amplio rango de entornos de desarrollo como son Microsoft Visual C++, C#, VB.NET, VB6, Java, Delphi y C++Builder haciendo que el SDK sea muy versátil en cuanto a posibilidades de diseño. Entre sus características destacan la detección de cara, de ojos y 66 rasgos faciales así como reconocimiento facial del que afirman que consiguen tasas de reconocimiento del 85 % cuando la tasa de falsa aceptación se establece en un 0,1 % para la base de datos FERET. Finalmente, ofrece funciones de paralelización con el objetivo de mejorar el rendimiento.

- Disponible en: <http://www.luxand.com/facesdk/>

3

Sistemas desarrollados.

En este capítulo se describen en detalle los dos sistemas diseñados, desarrollados e implementados en el proyecto:

- **Sistema off-line.** Este primer sistema desarrollado es un sistema de reconocimiento facial completo capaz de trabajar con bases de datos y producir resultados evaluables.
- **Sistema en tiempo real.** El segundo sistema tomará parte de las etapas del primero y se adaptará al funcionamiento en tiempo real mediante la captura de imágenes a través de una cámara web. Este sistema cuenta además con una interfaz, haciéndolo más fácil y configurable durante su manejo.

3.1. Sistema off-line

3.1.1. Descripción

El objetivo del primer sistema es el de encapsular todas las etapas de un sistema de reconocimiento facial desde la detección inicial hasta la decisión final. Este sistema, por tanto, obtendrá una serie de imágenes de entrenamiento y test sin procesar para hacerlas pasar por todas las etapas tal y como muestra el diagrama de la Figura 3.1.

El sistema es completamente modular, por lo que se pueden intercambiar fácilmente diferentes técnicas en cada una de las etapas para comparar resultados. Esta modularidad es además muy importante para la implementación del segundo sistema, ya que en su funcionamiento interno utilizará todas las funciones de este primer sistema.

Para realizar medidas de rendimiento de las técnicas implementadas, este sistema generará a su salida un archivo con las puntuaciones correspondientes a la comparación de cada imagen de test respecto a un conjunto de entrenamiento.

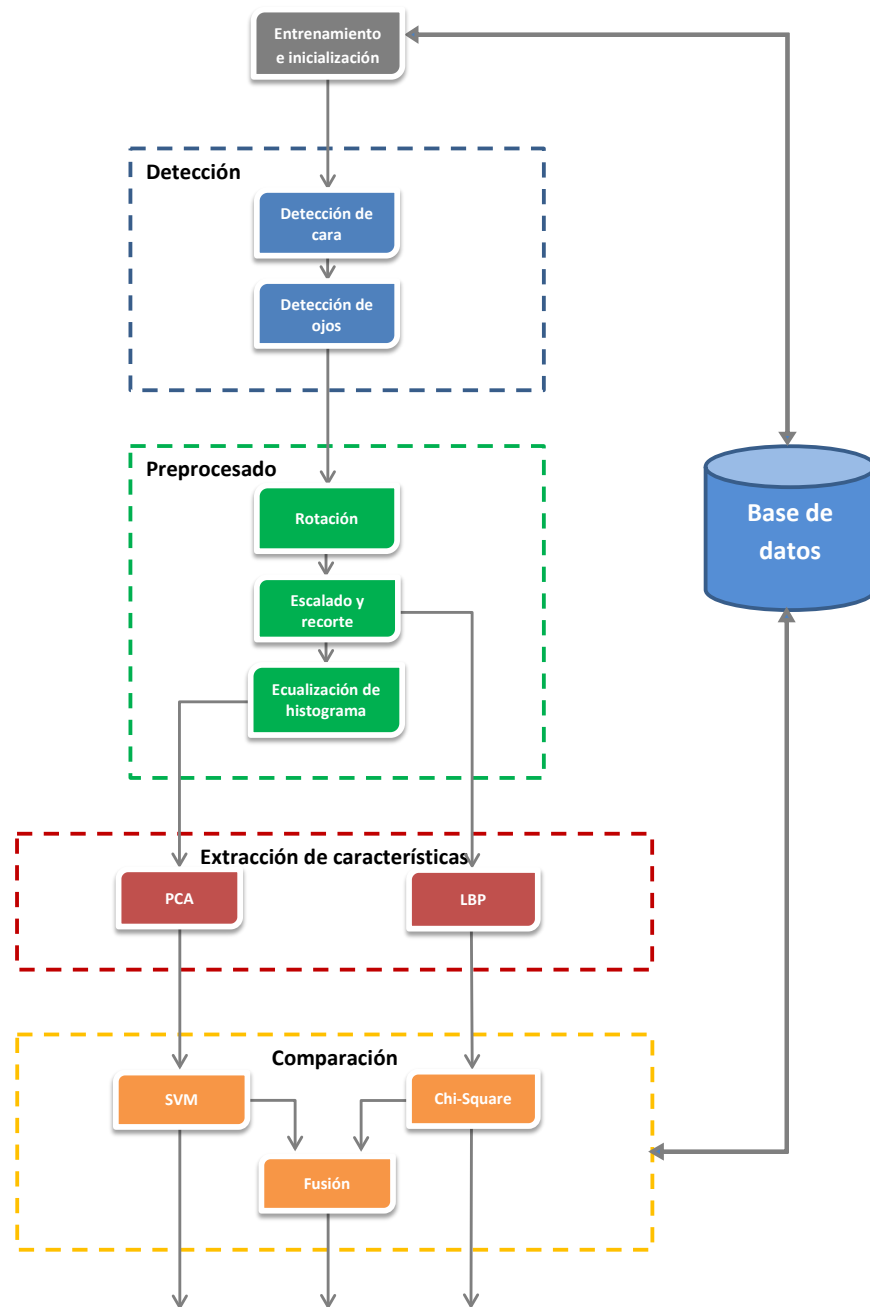


Figura 3.1: Diagrama de estados del sistema off-line.

3.1.2. Entrenamiento e inicialización del sistema

Esta etapa inicial del sistema difiere del resto puesto que sólo se ejecuta una vez durante todo el proceso, mientras que las consecuentes etapas se van repitiendo cada vez que entra una imagen de test en el sistema.

En esta etapa se leen los archivos que contienen las rutas de la base de datos de entrenamiento y de test y se aplica la correspondiente transformación a las imágenes dependiendo del modo de funcionamiento. Por ejemplo, si el sistema está utilizando técnicas de PCA para la reducción de dimensionalidad y SVM para la clasificación[30], durante la inicialización se obtendrán las imágenes de entrenamiento donde un subconjunto será utilizado para entrenar el PCA y el resto serán las imágenes que se transforman al subespacio PCA, obteniendo una serie de vectores que, a su vez, son utilizados para entrenar los SVMs. Como el sistema trata de acercarse lo máximo posible a un funcionamiento en tiempo real, en este proceso inicial no se transforman las imágenes de test puesto que son imágenes de las cuales no se tendría información hasta el mismo momento de su captura.

Tal y como se muestra en la Figura 3.2, las imágenes de test son imágenes sin ningún tipo de procesamiento, mientras que las de entrenamiento son imágenes de caras ya normalizadas pertenecientes a un individuo tal y como se hubieran realizado en un entrenamiento previo.



Figura 3.2: Ejemplos de a) imagen de test y b) imagen de entrenamiento.

Por último, para el correcto funcionamiento del algoritmo de Viola-Jones, se carga una cascada preentrenada.

3.1.3. Detección

La primera etapa por la que pasa la imagen de test una vez ha sido capturada es la etapa de detección. La información que se necesita detectar para las siguientes etapas son la cara y la posición de los ojos. En el caso de detección de la cara se utiliza el algoritmo de Viola-Jones [31] puesto que da muy buenos resultados. En el caso de la detección de ojos, se han implementado dos técnicas con diferentes características expuestas más adelante.

Detección de la cara

La detección de la cara es un aspecto crucial en el sistema puesto que un error al detectar la misma resultará en un error en el resto de etapas. Para esta función, se ha implementado detección de la cara mediante el algoritmo de Viola-Jones por su bajo coste computacional respecto a otras técnicas y su precisión en la detección.

El algoritmo de Viola-Jones fue desarrollado a partir de la obtención de sets de características basadas en *Haar Wavelets*, reduciendo significativamente el coste computacional que las técnicas basadas en intensidades de las imágenes suelen tener. Dicho algoritmo hizo posible la detección de objetos en tiempo real obteniéndose el primer detector de caras en tiempo real a través del mismo. El coste computacional, aunque es mucho menor que otras técnicas, sigue siendo significativo en aplicaciones en tiempo real. Por esta razón, el sistema sólo detecta una cara (la de mayor tamaño) en cada imagen de test dado que detectar múltiples caras hace que el sistema se ralentice de sobremanera, algo que se pretende evitar en la medida de lo posible para un correcto funcionamiento en tiempo real.

OpenCV incluye una implementación de detección de objetos utilizando este algoritmo [32] (ver Figura 3.3). Con esta función es posible variar el tamaño de la ventana mínima de detección e imponer un escalado a la propia imagen para reducir los tiempos de cómputo. La detección de múltiples objetos no está del todo optimizada y requiere un tiempo diez veces mayor al de la detección de un solo objeto (aunque sólo haya dos caras en la imagen).

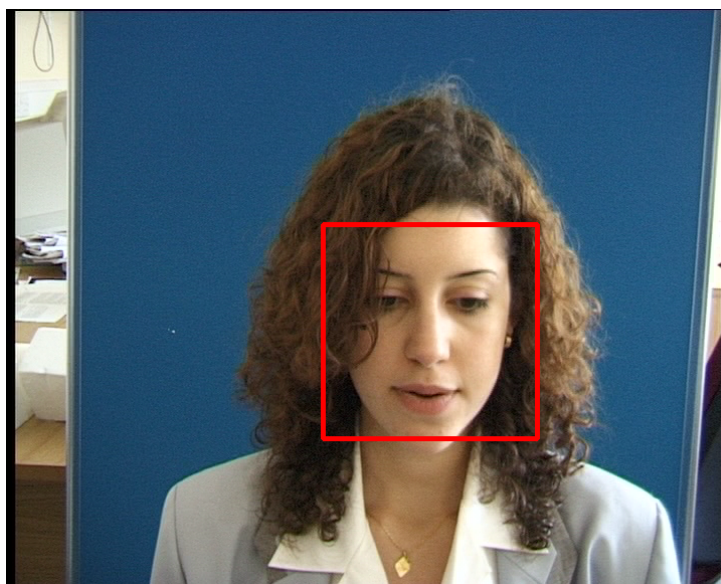


Figura 3.3: Ejemplo de detección facial usando el algoritmo Viola-Jones.

Detección de los ojos

Una vez detectada la cara, el siguiente paso es detectar la posición de los ojos para así obtener sus coordenadas y poder normalizar la imagen respecto a dicha información.

Se han propuesto dos técnicas: la utilización del algoritmo de Viola-Jones [33], al igual que con la detección de la cara, y la utilización de la información de la imagen binarizada para estimar la localización de los ojos [11]. En ambas técnicas y para facilitar la detección, la región

extraída de la cara es reducida aún más puesto que por geometría facial se puede afirmar que los ojos no se van a encontrar en la parte inferior de la misma o en la zona más superior. En la Figura 3.4 se puede ver las región acotadas para la detección de ojos.

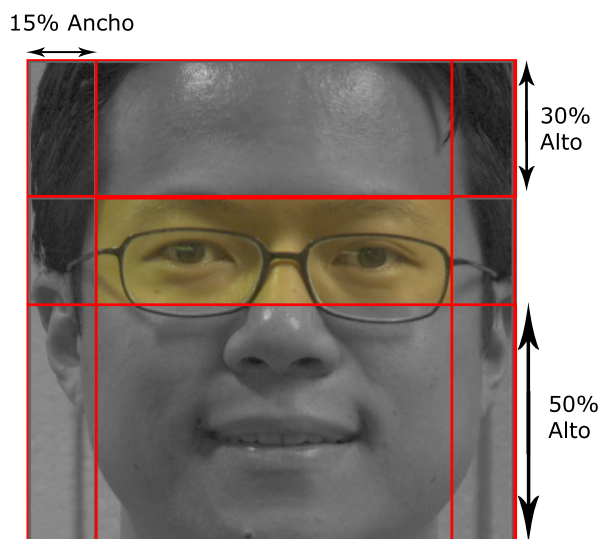


Figura 3.4: Región acotada para la detección de ojos.

El algoritmo de Viola-Jones se ejecuta de forma similar que en la detección de caras con la diferencia de que los clasificadores han sido entrenados esta vez con imágenes de ojos (un clasificador diferente para cada ojo). La imagen de la cual se van a detectar los ojos es sólo la cara previamente detectada por lo que, al reducir considerablemente el tamaño de la misma, se reduce a su vez el tiempo de procesado. La detección se realiza para cada ojo por separado y, una vez se obtiene la región del ojo, se procede a estimar por geometría la localización del mismo. Con este método se obtiene una detección bastante precisa del ojo siendo además robusto frente a ojos que no miran al frente (la pupila no modifica la localización de la mitad del ojo).

Por otra parte, la técnica de binarizar la imagen y buscar máximos de píxeles a negro no otorga tanta precisión pero da mejores resultados en imágenes de baja calidad puesto que, en imágenes donde el ojo se encuentra muy deteriorado, el anterior algoritmo simplemente no encontraría similitudes y no sería capaz de localizar el ojo, mientras que esta técnica sí que daría una estimación de la localización.

Para aplicar esta técnica se realiza el siguiente procedimiento; se binariza la imagen mediante el método de Otsu puesto que es capaz de encontrar automáticamente un umbral de binarización óptimo para cada imagen [34]. Una vez se tiene la imagen binarizada, se divide horizontalmente en dos mitades que serán estudiadas por separado para encontrar los ojos. Se suma el número de píxeles que están a negro en cada columna, creándose un histograma del cual se busca su pico máximo correspondiente con la posición de la pupila horizontalmente. Análogamente, se suman los píxeles a negro en cada fila de la imagen para buscar el pico máximo que se corresponderá con la posición de la pupila en el eje vertical.

Implementando este procedimiento surgen varios problemas relacionados con la aparición de falsos picos máximos. El primer caso habitual es la existencia de las cejas que crean un pico máximo cuando se estudia la imagen por filas. Para solventar este problema, se obtienen los dos

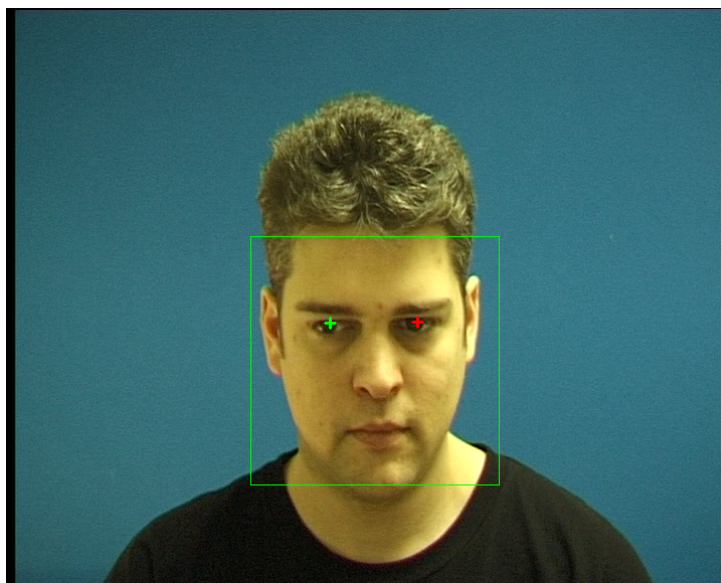


Figura 3.5: Ejemplos de detección de ojos mediante Viola-Jones.

picos mayores de la imagen y se escoge siempre el inferior que debería corresponderse con la pupila. El segundo caso de fallo es el producido por la existencia de cabello en los laterales de la imagen. Reduciendo la región de detección se minimiza este fallo pero no se llega a neutralizar puesto que en ciertas imágenes de personas con pelo largo, este queda dentro de la imagen.

Cabe destacar que, en muchas imágenes, la zona de los ojos queda ligeramente oscurecida por la sombra que proyectan las cejas haciendo que la binarización no sea óptima (Figura 3.6 b)) y aparezcan grandes regiones de negro en la zona del lagrimal (zona de mayor profundidad) produciendo falsos picos.



Figura 3.6: a) Binarización correcta. b) Binarización propensa a fallos.

3.1.4. Preprocesado (Normalización)

En la etapa de preprocesado se llevan a cabo las operaciones de rotación, escalado y recorte de la imagen de la cara con el objetivo de normalizarla al estándar propuesto por la norma ISO/IEC 19794-5 [18]. Al normalizar las imágenes, sólo la región de la cara queda visible, eliminando orejas, cabello y barbilla. Asimismo, todas tendrán el mismo tamaño (168x192) y coordenadas de los ojos (i.e., una distancia entre centros de los ojos de $IPD = 96pxeles$) para que sea posible la comparación entre ellas.

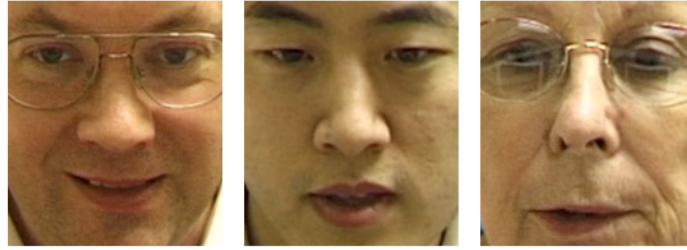


Figura 3.7: Imágenes normalizadas.

Rotación

La primera transformación que se lleva a cabo en la etapa es la de rotación de la imagen. Utilizando la información de las coordenadas de los centros de los ojos obtenidas en la etapa de detección, es posible calcular el ángulo de giro de la cara detectada respecto al eje horizontal y compensarlo mediante una transformación afín de rotación (ver Figura 3.8). En esta transformación se toma como centro el ojo derecho, puesto que será la referencia del resto de transformaciones.



Figura 3.8: Rotación de la imagen.

Escalado y recorte

Una vez se tiene la imagen rotada y nivelada, se procede a realizar una transformación geométrica de escalado en la misma para que la separación entre los ojos cumpla con el estándar. Una vez más, con las coordenadas actualizadas de los ojos se calcula la distancia entre los mismos y se obtiene una relación de proporción respecto al tamaño deseado. Como método de interpolación, la función de escalado de OpenCV utiliza interpolación bilineal.

Por último, utilizando como referencia las coordenadas del ojo derecho y una ventana de tamaño 168x192 colocada a una distancia del mismo como dicta la Figura 3.7, se recorta para obtener la imagen normalizada final. Como las funciones de recorte de OpenCV no autorellenan la imagen si alguna zona del rectángulo queda fuera, hay que asegurarse de añadir un marco negro a la misma ya que de lo contrario OpenCV da un error relativo a acceso de memoria inválida (los píxeles que quedan fuera de la imagen y que por consiguiente no apuntan a zonas válidas de memoria) provocando que el programa termine de forma incorrecta.

3.1.5. Extracción de características

Para obtener mejores resultados, la etapa de extracción de características se encarga principalmente de quedarse con los valores que realmente dan información de cara al reconocimiento y desechar aquellos valores que no aportan información relevante o que incluso pueden introducir información que haga más difícil el reconocimiento entre usuarios. Con esto también se pretende reducir la dimensionalidad de los vectores de características para reducir la carga computacional del reconocimiento.

En esta etapa se han implementado dos técnicas diferentes de extracción de características. Una de ellas es una técnica global tradicional muy extendida llamada *Principal Component Analysis* (PCA) y la otra es una técnica local llamada *Local Binary Patterns* (LBP) que destaca en su rapidez de cómputo.

PCA - *Eigenfaces*

Como se ha explicado anteriormente, PCA pretende reducir la información quedándose con un conjunto llamado componentes principales capaces de representar mejor los datos en términos de mínimos cuadrados. Para ello, proyecta los datos en un subespacio de coordenadas calculado mediante el entrenamiento previo con imágenes de caras. Durante dicho entrenamiento se obtiene una serie de autovalores y autovectores que realizarán la transformación de las imágenes de entrada. Las componentes principales quedan ordenadas en función de la información que contienen por lo que se puede comprobar que a partir de un cierto número de componentes, la información que se aporta es mínima.

La aplicación de PCA en imágenes de caras se conoce comúnmente como *Eigenfaces* ya que al representar los autovectores, la imagen resultante tiene la apariencia de ser una media de las caras entrenadas como se muestra en la Figura 2.10. Para el sistema, se utilizan 200 componentes principales para describir cada imagen ya que tal y como muestra la Figura 3.9, la pérdida de información no es realmente significativa.

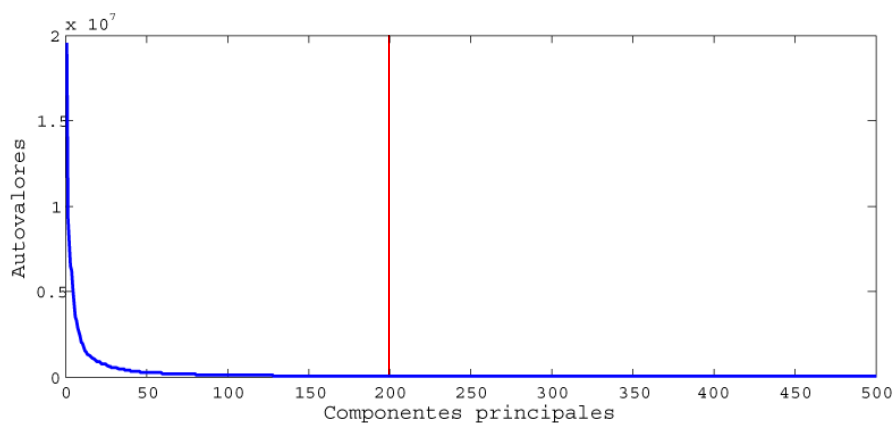


Figura 3.9: Punto de selección de 200 componentes principales una vez entrenado el PCA.

El procedimiento para proyectar una imagen de entrada en el subespacio PCA es el siguiente:

- **Transformación a escala de grises.** Si la imagen de entrada es a color, es necesario transformarla a escala de grises ya que PCA sólo trabaja con un canal. La función de

conversión de color a escala de grises de OpenCV realiza el siguiente cálculo en cada píxel: $Y \leftarrow 0,299 \cdot R + 0,587 \cdot G + 0,114 \cdot B$ ya que el ojo humano es más sensible a los espectros verde y (en menor medida) rojo.

- **Ecualización de histograma.** Debido a que el método de PCA no es robusto frente a cambios de iluminación, una forma de minimizar estas variaciones es ecualizando el histograma de la imagen. El objetivo de ecualizar el histograma en una imagen es el de que los valores de los píxeles ocupen todo el rango de 0 a 255 de la forma más constante posible. Con esto se consigue que no existan imágenes muy oscuras o muy claras y que todas tengan un mayor contraste (ver Figura 3.10). Este procesado no se lleva a cabo en la anterior etapa puesto que otras técnicas como LBP no requieren esta ecualización.

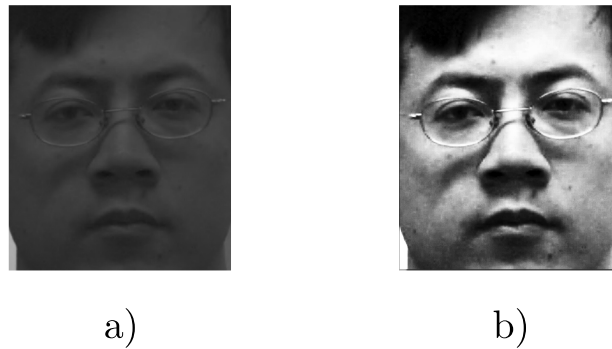


Figura 3.10: Imagen a) antes y b) después de ecualizar su histograma.

- **Transformación a vector columna.** La imagen de M filas y N columnas se transforma a un vector columna de $M \times N$ filas para que pueda ser proyectado en el subespacio PCA. La transformación es una simple transposición de filas a columnas en orden descendente añadiéndose a la misma columna.
- **Proyección en el espacio PCA.** Una vez que se ha obtenido el vector columna de la imagen, este se multiplica por la matriz de transformación PCA previamente entrenada. La matriz de transformación tiene K filas correspondientes al número de Componentes Principales establecido y $M \times N$ columnas. Al realizar la multiplicación de matrices, se obtiene un vector columna de Componentes Principales de dimensión K .

LBP histograms

La segunda técnica de extracción de características implementada es la basada en histogramas de *Local Binary Patterns*. LBP es un algoritmo bastante sencillo que transforma las imágenes sin necesidad de un entrenamiento previo como ocurre con PCA.

El funcionamiento básico de LBP es el siguiente; se recorre la imagen píxel a píxel y en cada posición se obtiene tanto el valor del píxel estudiado como de su vecindario. Los valores del vecindario son binarizados tomando como umbral el valor del píxel estudiado tal y como muestra la Figura 3.11. Una vez hecho esto, el vecindario será una sucesión de 8 cifras a uno y a cero, esta cadena se toma a partir del valor central izquierdo en sentido antihorario creando un

número de 8 bits o lo que es lo mismo, un número decimal de 256 valores. Este nuevo número extraído es sustituido por el píxel estudiado. Por último se salta al siguiente píxel de la imagen para repetir el proceso.

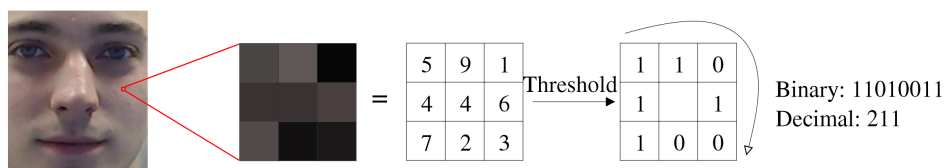


Figura 3.11: Operador LBP de radio 1.

La imagen resultante tiene aspecto de una imagen en blanco y negro que retiene los detalles de la imagen como muestra la Figura 3.12.



Figura 3.12: Imágenes transformadas mediante el operador LBP.

Este operador LBP fue posteriormente extendido para poder trabajar con vecindarios de radio mayores que uno y con un número de bits diferente a ocho. Cuando se utiliza un radio mayor que uno, se utiliza interpolación bilineal cuando el punto de muestreo no está en el centro de un píxel. Utilizar un radio superior produce un efecto similar al suavizado en la imagen transformada. Utilizar menos o más puntos de muestreo resultará en que una imagen tenga valores menores o mayores que 256, pudiendo variar el tamaño de los histogramas así como la información que contienen.

Para la generación de histogramas, la imagen transformada se divide en regiones y se calcula el histograma por separado de cada una. Finalmente todos los histogramas se concatenan para crear el vector de características (ver Figura 3.13). Dicho vector contendrá información sobre los valores de la imagen así como información local debido a la división en regiones.

Las regiones en las que se divide la imagen no tienen por que ser del mismo tamaño ni ser de alguna forma en concreto, pueden incluso solaparse entre ellas. Cuantas más regiones tenga la imagen, mayor será la cantidad de información, pero mayor será a su vez el vector de características. Para reducir aún más la dimensionalidad del vector se propuso un método basado en los llamados *Uniform Patterns*[24], que parte de la premisa de que las cadenas binarias calculadas en imágenes de caras que tienen más de dos cambios de uno a cero y viceversa solo contabilizan un 10 % del total de valores por lo que agruparlos en un solo valor del histograma no conllevaría una pérdida de información muy significativa y, sin embargo, reduciría en gran medida el tamaño de los histogramas puesto que de los 256 valores, sólo 58 cumplen la característica de *Uniform Pattern*. De esta forma los histogramas tendrán un tamaño total de 59 valores (58 *Uniform Patterns* + 1 para el resto).

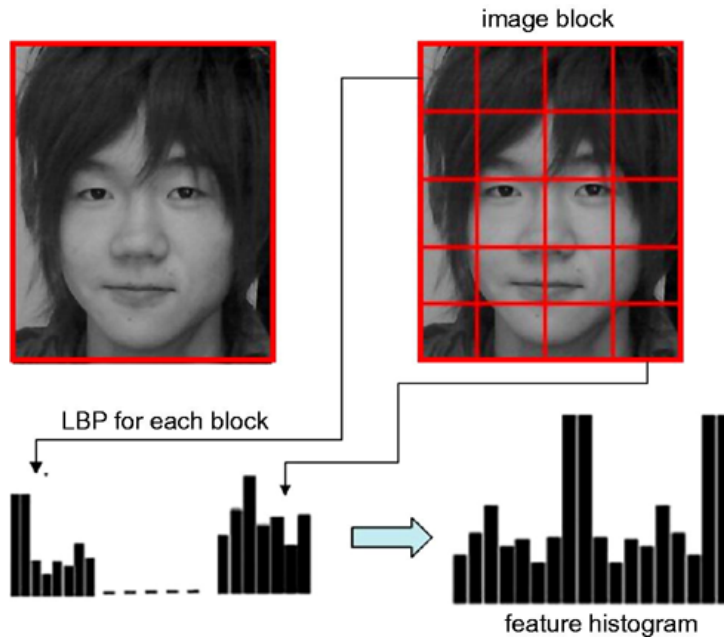


Figura 3.13: Cálculo de los histogramas LBP de cada región.

En este sistema se utilizan 8 puntos de muestreo y un radio de vecindario de 2 píxeles puesto que es la configuración que comúnmente mejores resultados ofrece[23]. Las imágenes han sido divididas por una cuadrícula de 7x7 rectángulos totalizando 49 regiones de mismo tamaño. Además se ha implementado el uso de *Uniform Patterns* para reducir el tamaño de los histogramas teniendo 59 valores por cada región. Cabe destacar que **OpenCV no ofrece ninguna función para utilizar LBPs** por lo que se han tenido que implementar diversas funciones modificadas [35] para obtener los resultados esperados, proporcionando una implementación propietaria.

3.1.6. Clasificación y decisión

La última etapa del sistema comprende la clasificación de la información y la obtención de una decisión mediante medidas de distancia. Dependiendo del método de extracción de características que se haya utilizado, se han implementado técnicas que dan buenos resultados con los mismos. Para PCA, se ha implementado una medida de distancia básica para realizar las pruebas iniciales como es la distancia euclídea y, por otro lado, un clasificador más lento y a la vez más avanzado como son los SVMs. Para el método de LBP se utiliza como medida la distancia χ^2 (Chi-Square) tal y como se recomienda en la literatura [24].

Distancia Euclídea

En el caso del sistema utilizando PCA, en vez de tener dos dimensiones como se ejemplificó en el estado del arte, se tendrían doscientas pero el cálculo se realizaría de la misma forma. Una vez se obtiene la distancia, un valor menor indicará que los vectores de características están más cerca el uno del otro o, particularizando, que dos imágenes de cara tendrán mayor parecido.

SVM

La segunda técnica implementada para el sistema funcionando con PCA es el uso de SVMs para realizar la clasificación. Una desventaja de los mismos frente a las técnicas de medidas de similitud es que requieren un entrenamiento previo para poder realizar clasificaciones pero suelen obtener mejores resultados.

Para el caso del sistema, en vez de entrenar un SVM con todas las clases y obtener la clase en la que se encuentra cada dato de test que entra, se ha optado por entrenar un SVM por cada imagen de entrenamiento para poder obtener una medida de distancia [30]. Para entrenar cada SVM, se toma el vector de características de la imagen de entrenamiento como una clase y el resto de vectores de las imágenes en la base de datos como otra clase, teniendo sólo dos clases. Con esto se obtiene la distancia que se tiene a cada imagen por separado. Dado que **OpenCV no tiene implementado este tipo de funcionamiento en SVMs** [36], se ha tenido que modificar el código de la función que trae la librería para poder acceder a los parámetros internos. El kernel utilizado para el cálculo del hiperplano es lineal, este kernel dará peores resultados que los no lineales sin embargo estos últimos son mucho más lentos por lo que los hacen poco viables para una aplicación en tiempo real.

Chi-Square

Para el caso de los LBPs, la utilización en conjunto con la medida de distancia χ^2 obtiene buenos resultados. Aunque no tiene la capacidad de clasificación de los SVMs, χ^2 se calcula de una forma muy rápida y sin necesidad de entrenamiento previo por lo que, unido a la rapidez de cálculo de los LBPs, lo convierten en un método interesante para los sistemas en tiempo real.

3.2. Sistema en tiempo real

3.2.1. Descripción

El sistema en tiempo real se basa principalmente en el funcionamiento del sistema anteriormente descrito con la diferencia de que se han añadido, modificado y rediseñado nuevas funcionalidades como la capacidad de entrenar individuos o la de detectar múltiples caras en una imagen. Adicionalmente, se ha diseñado una interfaz con la que se puede controlar las diferentes funciones del programa como son entre otras, la selección del modo de funcionamiento o la elección de resolución de captura. Tal y como muestra la Figura 3.14, la captura de imágenes se realiza a través de una cámara web y, dependiendo del modo en el que se esté trabajando, el sistema pasa por etapas diferentes marcadas en caminos de colores en el diagrama (las cajas con bordes blancos corresponden al sistema off-line y las que tienen bordes coloreados al nuevo sistema).

Este sistema está integrado con Qt, un entorno de desarrollo de aplicaciones que ofrece gran cantidad de funcionalidades respecto a la creación de interfaces gráficas. Esta decisión se ha tomado debido a que OpenCV está muy limitado respecto a elementos de interfaz gráfica. Los únicos elementos dinámicos que ofrece son sliders, por lo que el desarrollo de una aplicación compleja se convierte en una tarea imposible.

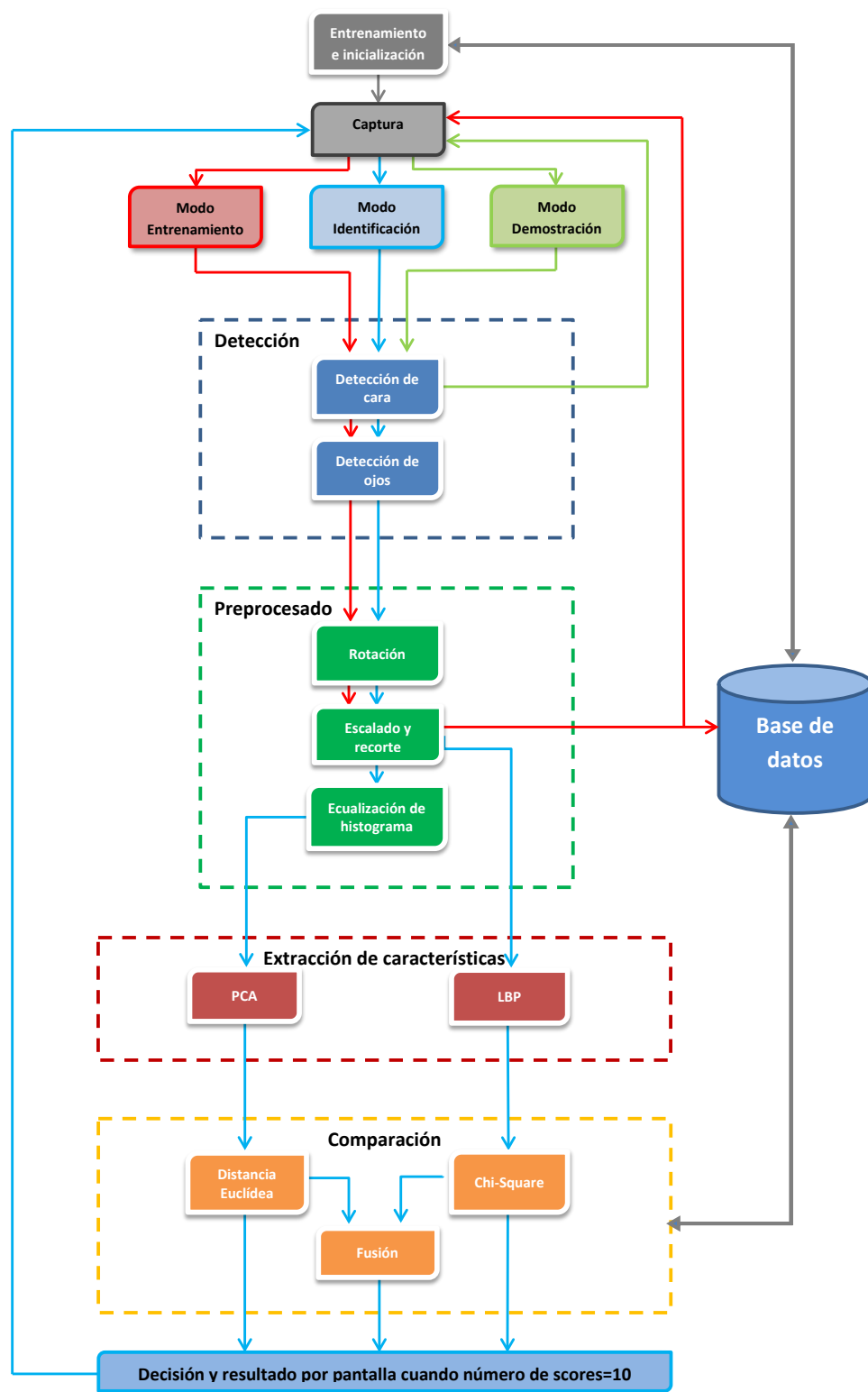


Figura 3.14: Diagrama de etapas del sistema en tiempo real.

3.2.2. Equipamiento utilizado

Una parte esencial del sistema es el equipamiento que se utiliza puesto que las especificaciones del mismo afectan de forma directa al rendimiento tanto en tiempo como en precisión. A continuación se describen los tres elementos de hardware que componen el sistema montado en el laboratorio.

- **Ordenador de sobremesa.** El equipo encargado de realizar las tareas principales de procesamiento tiene instalado los siguientes componentes:
 - **Procesador.** Intel Core 2 Duo E6700. Frecuencia de reloj 2,66 GHz. 4 MB Caché L2. 2 núcleos físicos.
 - **Memoria RAM.** Dos pastillas de 2048 MB a doble canal. Frecuencia de trabajo 400 MHz. Tipo de memoria DDR2.
 - **Disco Duro.** Capacidad de 500 GB. Velocidad 7.200 Rpm.
 - **Tarjeta Gráfica.** NVIDIA GeForce GTX 560 Ti 1 GB.
 - **Sistema Operativo.** Windows 8 64 bits.

El ordenador influirá en la velocidad de procesamiento del sistema y, por tanto, en el tiempo total que requiera desde que entra una imagen hasta que captura la siguiente viéndose afectada directamente la tasa de fotogramas por segundo.

- **Cámara web.** La cámara web encargada de capturar las imágenes es una Logitech C920 capaz de grabar vídeo en Full HD (1080p). También es capaz de grabar vídeo en otras resoluciones por lo que se aprovecha esto para que el programa pueda trabajar con diferentes resoluciones. La tasa de fotogramas por segundo máxima es de 30, por lo que el sistema se verá limitado por esta especificación. Las diferentes resoluciones se traducen en imágenes de mejor calidad por lo que cabe esperar que con una mayor resolución, el sistema funcione de manera más precisa (siempre y cuando el entrenamiento se realice también con resoluciones altas) respecto a resoluciones más bajas, sobre todo en distancias alejadas. Cabe añadir que una mayor resolución también contribuye a que el sistema se ralentice porque las imágenes son de mayor tamaño.
- **Televisor.** El televisor por donde se visualiza el resultado del programa es un televisor THOMPSON de 47 pulgadas. La pantalla tiene una frecuencia de refresco de 200 Hz. Como muestra la Figura 3.15, dicha pantalla está montada sobre un soporte con ruedas que permite su movimiento por el laboratorio para trabajar en diferentes entornos.

3.2.3. Adaptación e implementación del sistema off-line para su funcionamiento en tiempo real

Como se ha explicado anteriormente, el sistema off-line ha sido usado en el segundo sistema para realizar las tareas de reconocimiento. Al estar este último programado en el entorno Qt, es necesario realizar algunos cambios antes de realizar la adaptación al funcionamiento en tiempo real.

La versión utilizada de Qt es la 5.0.1 siendo el Qt Creator (entorno de programación) la versión 2.6.2, mientras que con la versión de OpenCV se intentó utilizar la misma con la que se había programado el sistema offline, la última versión en su día 2.4.2. Tras configurar la librería



Figura 3.15: Equipamiento completo montado en el laboratorio.

y el entorno, se descubrieron errores en el funcionamiento muy posiblemente generados por incompatibilidades entre los dos módulos como por ejemplo fallos al realizar multiplicaciones entre matrices que impedían el correcto funcionamiento del sistema off-line. Tras múltiples pruebas, se actualizó la versión de OpenCV a la 2.4.5 y se configuró con Qt resolviéndose así todos los problemas de incompatibilidad. Es por ello que **se recomienda el uso de estas versiones para el desarrollo de aplicaciones.**

En relación con el diseño del sistema en tiempo real, las mismas funciones se utilizaron pero todos los tipos de variable de OpenCV *IplImage* fueron cambiados en casi su totalidad por el tipo *Mat* puesto que ofrece mejor gestión de la memoria. La adaptación más significativa es la captura a través de una cámara web en vez de la lectura a partir de una serie de archivos. Para lograr realizar esto dentro de la estructura de Qt, se establece un temporizador el cual cada 10 milisegundos carga una señal de la cámara y manda una señal a la función principal de captura. Esta función de captura se asemeja a la función *main* del sistema off-line puesto que dependiendo de los elementos de la interfaz que estén seleccionados, se encarga de realizar una función u otra a partir de llamadas a las funciones del sistema off-line. En resumen, una vez

se ha iniciado el sistema, se tiene una función la cual se ejecuta cada 10 ms si no hay alguna limitación (por ejemplo los 30 fps de la cámara web) y realiza las funciones pertinentes.

Para tratar de mejorar los tiempos de proceso en la etapa de detección y aprovechando las características del video, se propuso un método de diferencia de imágenes con el objetivo de eliminar el fondo y aplicar el algoritmo de detección sobre el área de interés. Este método fue descartado tras realizar pruebas debido a que no se conseguían resultados satisfactorios principalmente por la baja tasa de fotogramas por segundo a la que opera el sistema generándose regiones de movimiento incorrectas. Adicionalmente, cuando existía más de una región para el estudio, el algoritmo de Viola-Jones debía ser ejecutado para cada región y esto producía tiempos mayores que el algoritmo actuando sobre la imagen completa. No obstante, este método resulta interesante para acotar las regiones de procesado y se propone como trabajo futuro.

La lectura de los archivos de entrenamiento se realiza automáticamente desde la carpeta designada a ello en vez de leer de un archivo como en el primer sistema, de esta forma cualquier cambio que se realiza es actualizado automáticamente. Finalmente, en la última etapa no se genera un archivo de resultados sino que se genera una decisión a partir de los datos obtenidos y se muestra por pantalla. Puesto que el programa trabaja con fotogramas de vídeo, se aprovecha este entorno y la decisión final se obtiene después de haber capturado al menos 10 imágenes correctamente. Las puntuaciones de estas diez imágenes son promediadas para obtener una mejor estimación.

3.2.4. Desarrollo de la interfaz

Para el diseño de la interfaz, se ha tratado de evitar múltiples ventanas y tener toda la información en una sola ventana puesto que la pantalla es lo suficientemente grande para albergar múltiples datos. De esta forma todas las opciones están visibles y son de fácil acceso.

En la Figura 3.16 se puede observar una captura de pantalla de la interfaz del programa cuando se ejecuta por primera vez (los tamaños de letra han sido modificados para facilitar su visualización en este documento, el programa real puede presentar ligeras diferencias al respecto). La interfaz se divide en tres zonas principales. En la parte izquierda de la ventana se encuentran todos los elementos de control para hacer que el sistema funcione en diferentes modos. La zona central es el área principal donde se visualiza lo que captura la cámara web y, a su vez, se ofrece información de lo que se detecta y reconoce. Por último, en la derecha se visualizan elementos que ofrecen información adicional como los tiempos de procesado o la imagen normalizada resultante de la detección. A continuación se describe cada elemento de la interfaz marcado en la Figura 3.16 para, más adelante, detallar en profundidad las modos de funcionamiento.

1. **Inicializar utilizando técnicas PCA y Distancia Euclídea.** El primer elemento del menú pone en marcha el sistema y lo prepara para identificar individuos utilizando técnicas de PCA con distancia euclídea. Al hacer click en el botón, se llama a la función que busca la base de datos en el directorio especificado y realiza el entrenamiento de las imágenes. Una vez hecho esto, activa el temporizador y la cámara para que comience a capturar imágenes. Cuando el sistema termina de inicializar, los controles para identificar, detectar y entrenar se habilitan. Volver a pulsar el botón hace que se vuelva a inicializar el sistema con lo que se puede cambiar de técnica simplemente pulsando el botón deseado. Debido a que se tiene control absoluto sobre la base de datos, el PCA se entrena con las mismas imágenes de entrenamiento que se utilizan posteriormente en el reconocimiento.

2. **Inicializar utilizando técnicas LBP y χ^2 .** Similarmente al anterior botón, este inicializa el sistema para su funcionamiento utilizando LBPs con χ^2 . En este caso, las imágenes de la base de datos son transformadas a los histogramas de características generados por el método de LBP. Como no se requiere un entrenamiento previo, el tiempo necesario para inicializar este sistema es menor.
3. **Inicializar fusionando técnicas.** El último de los botones de inicialización prepara el sistema para funcionar fusionando las dos anteriores técnicas. El entrenamiento se realiza de la misma forma que en los dos casos anteriores juntos con la diferencia de que, al tratarse de una fusión de scores, el método de normalización de las mismas es la tangente hiperbólica [37]. Esto quiere decir que es necesario obtener la media y la desviación estándar de las puntuaciones entre las muestras de entrenamiento genuinas por lo que, adicionalmente al entrenamiento, se debe realizar el cálculo de scores haciendo que el tiempo para que el sistema esté preparado sea alto.
4. **Cambio de resolución.** Como la cámara web es compatible con varias resoluciones, este menú permite cambiar entre tres resoluciones diferentes. Las resoluciones disponibles son 640x480, 1280x720 y 1920x1080. El cambio entre las mismas se puede realizar durante cualquier función, incluso entrenando. La resolución por defecto que se ha elegido es 1280x720 puesto que ofrece buena calidad de imagen sin ralentizar de sobremanera el sistema. Con este elemento, se puede comprobar como afectan los cambios de resolución en el tiempo de procesado y en la precisión del sistema.
5. **Giro de la imagen.** Debido a que ciertas cámaras web (entre ellas la utilizada en el proyecto) capturan las imágenes con un giro de 180°, este botón se ha añadido al sistema para corregir este fenómeno.
6. **Identificación.** En este menú se puede activar la identificación. Pulsando el botón comienza el proceso haciendo uso del sistema off-line. Este funcionamiento es explicado con mayor detalle en la siguiente sección.
7. **Demostración de detección.** En este menú se controla la detección de múltiples caras. El número de caras a detectar se controla mediante el menú desplegable. El tipo de inicialización no afecta en este funcionamiento puesto que no llega a pasar de la detección.
8. **Entrenamiento.** El último modo de funcionamiento es el de entrenamiento de un individuo. En el campo de texto se introduce el nombre a entrenar y en el selector, el número de imágenes que se entrenarán de ese individuo (sin contar las que puedan haber ya entrenadas en la base de datos).
9. **Captura y resultado.** En esta zona se muestra la imagen que captura la cámara web. Cuando no hay ningún modo de funcionamiento activo, el programa simplemente muestra la captura, de lo contrario la imagen mostrada está procesada (por ejemplo, en detección se mostrarán cuadrados rojos sobre las caras detectadas).
10. **Versión e información.** Pulsando este botón, aparte de mostrar el número de versión del programa, se puede visualizar en una ventana emergente información sobre el autor del programa así como los entornos utilizados y las organizaciones involucradas.
11. **Imagen normalizada.** En esta pequeña zona se muestra la imagen normalizada siempre que la detección haya sido correcta. De esta forma, se tiene información visual si el sistema está siendo capaz de detectar con éxito tanto la cara como los ojos y cómo está normalizando los mismos.

12. **Tiempos de procesamiento.** En este último grupo de elementos se muestran los tiempos relativos al cómputo de las etapas más significativas. Dependiendo del modo de funcionamiento en el que se esté trabajando, los datos del área de tiempos cambiarán en consecuencia. Como últimos parámetros siempre se ofrece el total de tiempo en milisegundos que ha llevado realizar todas las funciones desde que entra la imagen hasta que termina la función principal y la tasa de fotogramas por segundo calculada a partir de esos datos. Cabe destacar que ciertas instrucciones internas no son contabilizadas puesto que el tiempo que tardan en procesarse se considera despreciable.

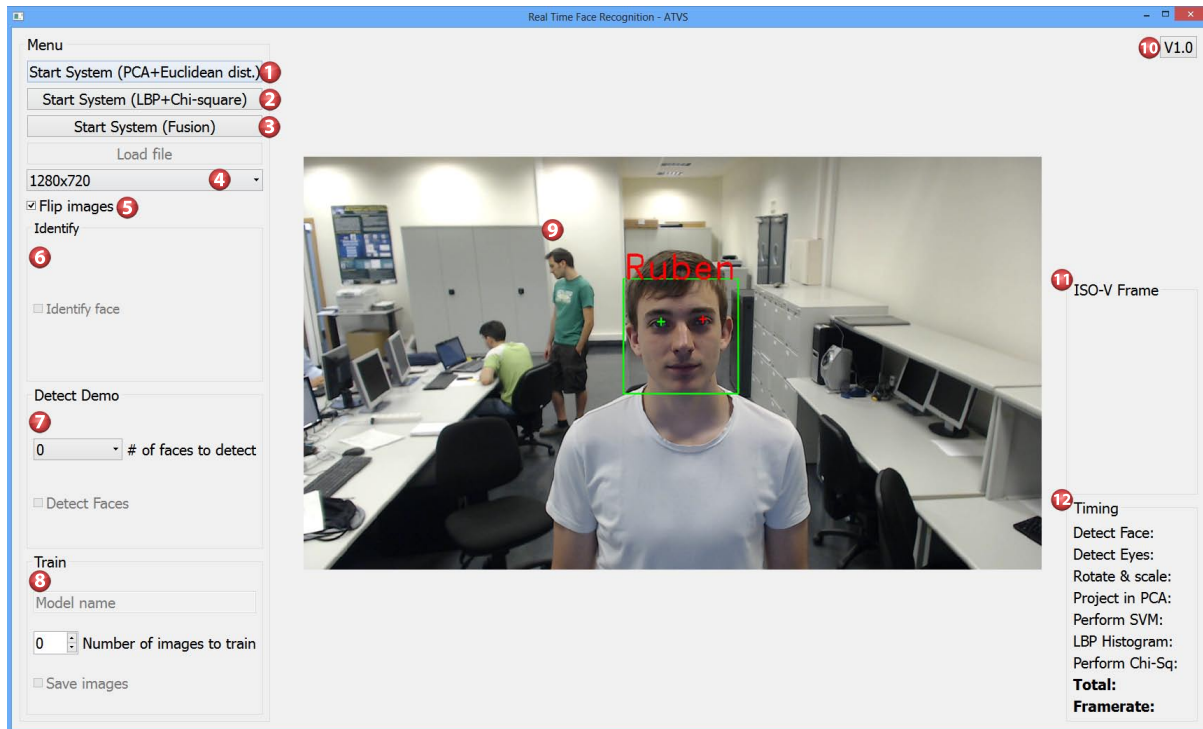


Figura 3.16: Interfaz del programa de reconocimiento en tiempo real.

La interfaz permite tres modos de utilización:

- **Modo 1. Identificación.** En este modo el sistema tratará de detectar y reconocer a los individuos que sean capturados por la cámara web.
- **Modo 2. Demostración.** Este modo muestra un detector de caras múltiple pudiendo seleccionarse el número de caras a detectar.
- **Modo 3. Entrenamiento.** El último modo permite entrenar usuarios para almacenarlos en la base de datos con un identificador asociado. El número de imágenes a entrenar es configurable.

Modo 1. Identificación

Uno de los modos principales de funcionamiento es el modo de identificación. Este modo es el más cercano al funcionamiento del sistema off-line haciendo uso de casi todas las funciones involucradas en el mismo. Cuando una imagen es detectada por la cámara web, comienza el

proceso de detección. Para que la detección se considere válida, debe haber detección tanto de la cara como de los dos ojos. Detectándose la cara, es posible estimar la posición de los ojos si su detección falla pero, aprovechando el hecho de estar manejando imágenes provenientes de una secuencia de vídeo, se decidió optar por desechar las que no son completamente detectadas para obtener una mayor precisión. En cada detección se marca la zona detectada y se muestra en el área principal de captura, en el caso de la cara con un recuadro y en el caso de los ojos con unas cruces, verde para el ojo derecho y roja para el izquierdo. Una vez se ha detectado una cara correctamente, se procede a realizar el preprocesado y, posteriormente, a mostrar la imagen normalizada en el área designada a ello (ver Figuras 3.17 y 3.18).



Figura 3.17: Sistema funcionando en modo identificación.

Dependiendo del tipo de inicialización que se seleccionase al comienzo, el programa realizará las conversiones pertinentes sobre la imagen normalizada para poder obtener una puntuación respecto a cada una de las imágenes almacenadas en la base de datos. Una vez más, aprovechando el funcionamiento con secuencias de vídeo, las puntuaciones obtenidas son almacenadas hasta obtener diez muestras válidas con sus correspondientes puntuaciones. Estas puntuaciones son promediadas para obtener unos valores más robustos frente a cambios repentinos como gestos faciales o posición de la cara. Para asegurar que la secuencia de imágenes contiene la misma cara durante la captura de puntuaciones, el sistema compara las posiciones anteriores de la misma y las compara con las capturadas. Si dicha posición varía en un número elevado de píxeles, se estima que la cara detectada es de otra persona y la contabilización de puntuaciones se reinicia. Este número de píxeles es variable dependiendo de la resolución que se aplique dado que a resoluciones mayores, el mismo área está definida por un número mayor de píxeles. Esto significa que, proporcionalmente, los cambios de posición en altas resoluciones generan una diferencia de píxeles mayor a tener en cuenta.

Una vez se tiene una decisión respecto a los datos obtenidos, el sistema muestra por pantalla el nombre de la persona identificada junto al recuadro de detección de la cara a la vez que este

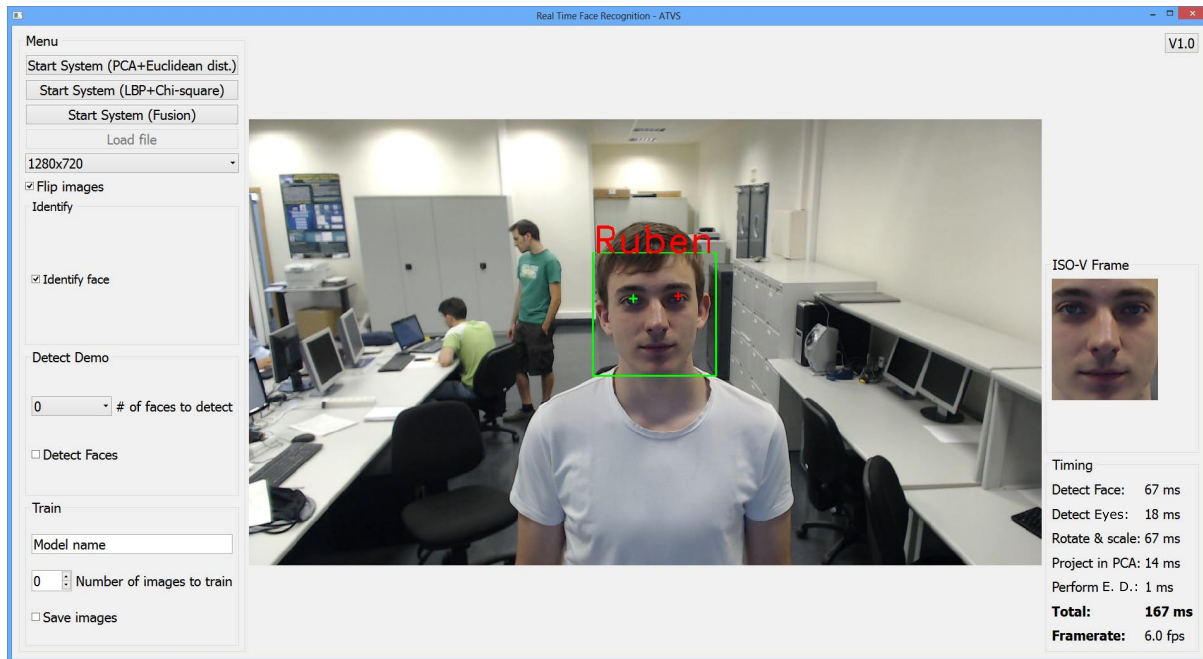


Figura 3.18: Detalle del sistema identificando correctamente a un usuario.

cambia a color verde, para darle un significado más visual. La decisión sobre una cara detectada no cambia a no ser que, por el método anteriormente definido, la cara detectada pase a ser la de otra persona.

Por último, hay que hacer notar que el sistema funcionando con PCA y SVM no ha sido posible ser implementado en tiempo real. El origen de este problema fue que el método para obtener las distancias de los SVMs tuvo que ser modificado del código fuente de las librerías de OpenCV, seguido de una recompilación de las mismas para su funcionamiento deseado. Si bien esto es funcional cuando se programa solamente con OpenCV, se descubrió que **Qt no se puede vincular correctamente cuando las librerías han sido modificadas**. Debido a este impedimento, se buscó una forma alternativa de extraer los parámetros de los SVMs que involucraban el uso de unas funciones de guardado de dichos parámetros en un archivo XML, que posteriormente se podía utilizar para leer los mismos de ahí. Esta solución fue también implementada pero se vio más tarde que la extracción de parámetros no se hacía correctamente, debido a un funcionamiento incorrecto del código fuente y a que los datos no se correspondían con los esperados. Todas estas dificultades hicieron imposible la correcta implementación de los SVMs en el sistema en tiempo real desarrollado.

Modo 2. Demostración

El modo demostración es un funcionamiento del sistema en el que sólo realiza detección de cara pero, a diferencia del sistema completo, existe la opción de detectar más de una cara cada fotograma. Las caras detectadas se ordenan en tamaño por lo que si, por ejemplo, se selecciona detección de tres caras y hay cinco posibles caras en la imagen, se marcaran con un recuadro rojo las tres caras más grandes que se hayan detectado. Un ejemplo de este funcionamiento se puede ver en la Figura 3.19.

OpenCV ofrece la posibilidad en su función de detectar una sola cara o varias caras [32]. El

detector múltiple está muy poco optimizado, puesto que llega a tener tiempos casi diez veces mayores que el detector de una cara. Estos tiempos de proceso no varían con el número de imágenes a detectar, permanecen constantes haya dos o diez caras en la imagen. También hay que señalar que cuando no hay ninguna cara en la imagen, el algoritmo tiene que realizar un barrido completo y los tiempos son muy elevados.

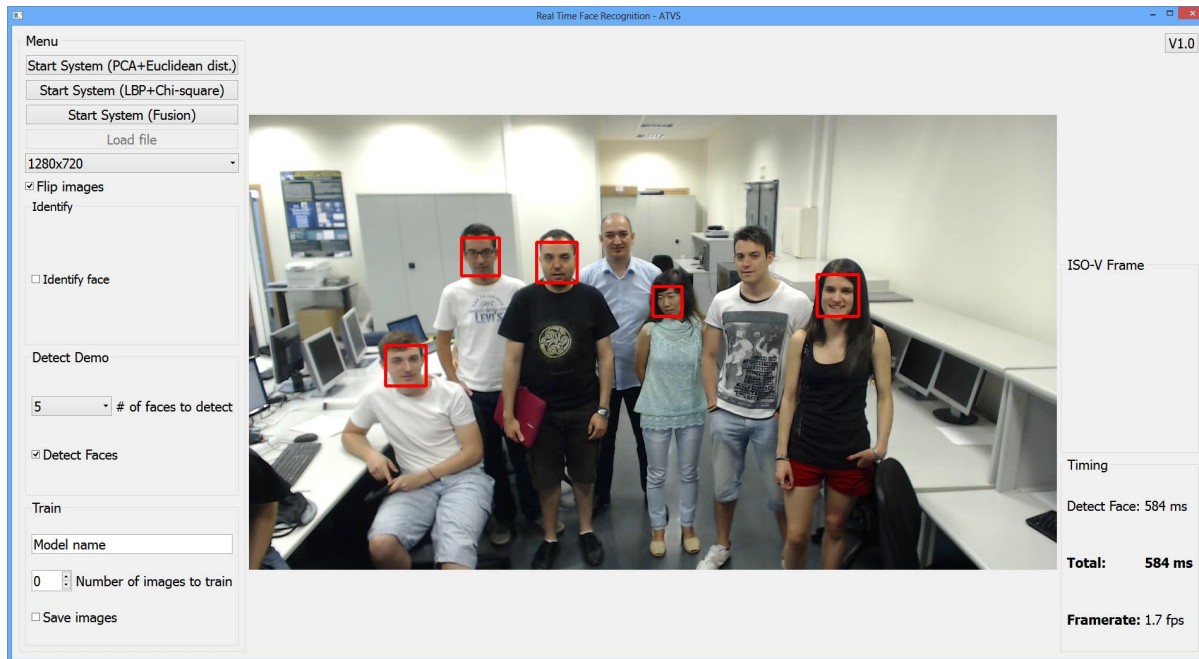


Figura 3.19: Sistema detectando múltiples caras.

Modo 3. Entrenamiento

El último modo de funcionamiento del que consta el sistema es el encargado de registrar a los usuarios y almacenar las imágenes de sus caras en la base de datos. En este modo se utilizan principalmente las funciones de detección de cara y ojos así como la de normalización. Para realizar el entrenamiento de un usuario, se introduce un identificador en el campo de texto como por ejemplo su nombre y se selecciona el número de imágenes que serán entrenadas y almacenadas. Si no se define un valor para el número de imágenes a entrenar, el sistema realizará un entrenamiento sin detenerse hasta que el usuario vuelva a hacer click en el botón de detección.

Una vez que el sistema inicia el entrenamiento, se detectará la cara del usuario y, en caso de éxito, los ojos. Si la detección de la cara o algún ojo no es correcta, la imagen no será procesada y almacenada. De esta forma se obtienen las caras con mayor precisión al igual que en la identificación. Visualmente en el área de detección se marcará la cara que está siendo entrenada con una etiqueta para conocer el estado. En el área de información adicional aparecerá la imagen normalizada una vez haya sido procesada, siendo esta misma la que se almacenará en la base de datos (ver Figura 3.20).

En la parte inferior donde se muestra la imagen normalizada, aparece un contador de el número de imágenes entrenadas en la sesión mientras dura el proceso. Este contador hace un recuento de las imágenes que se están entrenando para dar información sobre el estado, si existen más imágenes del usuario en la base de datos, estas no son contabilizadas. Como se puede ver en

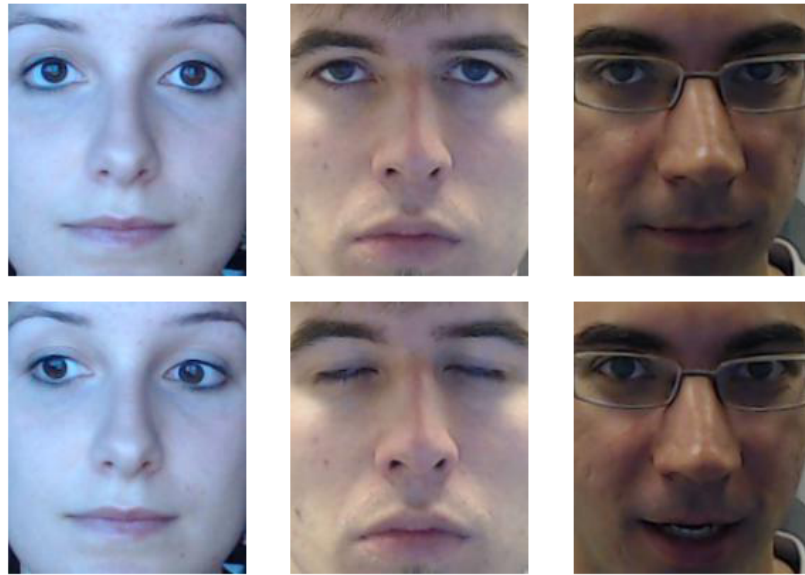


Figura 3.20: Ejemplo de imágenes capturadas y procesadas para entrenar el sistema.

la Figura 3.21, cuando se introduce un identificador en la base de datos, el sistema realiza una comprobación y, en caso de conflicto, avisa de que existe un usuario con el mismo identificador y muestra una imagen del mismo para que la persona controlando el programa compruebe si es el mismo o uno distinto. El sistema da dos opciones, una es no entrenar y cambiar el identificador porque no es la misma persona o la segunda que es entrenar porque es la misma persona. Si se elige la segunda opción, el sistema tomará la última imagen que haya entrenada y seguirá la numeración desde ese punto.

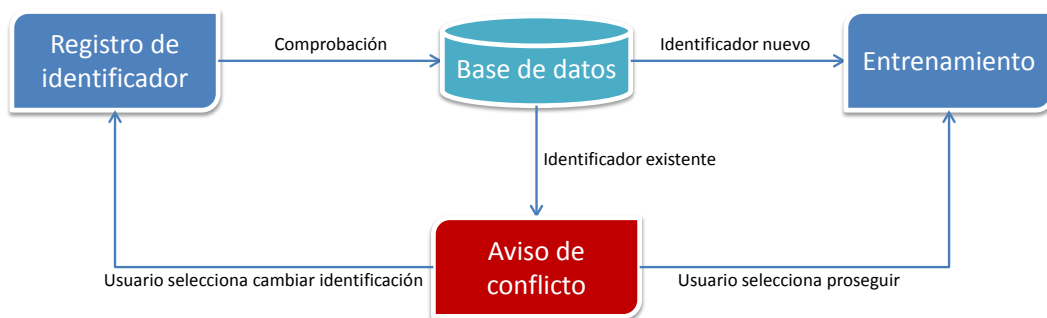


Figura 3.21: Diagrama de estados del funcionamiento de la comprobación de identificadores.

3.2.5. Generación de logs e información del sistema

Adicionalmente a la información que se muestra por pantalla durante el funcionamiento como el resultado de la imagen normalizada o los tiempos de procesado, el sistema también guarda en archivos información del funcionamiento. El sistema crea dos archivos, uno de ellos contiene información sobre el reconocimiento como puntuaciones y decisiones tomadas. El segundo archivo

guarda información relativa al rendimiento del sistema almacenando los tiempos de procesado requeridos en cada etapa.

En el archivo de puntuaciones, se va recogiendo el resultado de las puntuaciones de cada imagen de entrada respecto a las imágenes de la base de datos, así como el número de imágenes consecutivas de la misma para realizar la media una vez se llegue a diez. Cuando se ha realizado la media de puntuaciones y se ha tomado una decisión, en el archivo se añade un campo adicional mostrando las medias calculadas de las diez puntuaciones anteriormente registradas y la puntuación elegida, así como el identificador de usuario vinculado a dicha puntuación.

En el archivo de tiempos, se registran todos los tiempos correspondientes a las etapas que están funcionando. El contenido del archivo es prácticamente un volcado de la información que aparece en tiempo real por pantalla en el área de tiempos. Para una mejor visualización de los datos, se muestra también el modo de funcionamiento (e inicialización) correspondiente a los tiempos mostrados puesto que dependiendo del mismo, las etapas funcionando varían.

4

Experimentos realizados y resultados.

Los resultados experimentales centran en el análisis de 3 variables; i) análisis del rendimiento del sistema en detección, ii) análisis del rendimiento en reconocimiento y iii) análisis de los tiempos de procesado de cada etapa. Estos resultados serán tomados a partir de los sistemas desarrollados descritos en el capítulo anterior.

La siguiente sección describe las bases de datos BANCA[3] y FOCS[4] sobre las que se realizarán los experimentos junto con el protocolo experimental seguido.

4.1. Bases de datos y protocolo experimental

Las dos bases de datos utilizadas para la realización de los experimentos son: BANCA, creada por la Universidad de Surrey y Face and Ocular Challenge Series Database (FOCS) creada y utilizada para la evaluación NIST del mismo nombre[38].

4.1.1. Base de datos: BANCA

La base de datos de BANCA comprende la captura en tres escenarios diferentes de imágenes de cara de usuarios durante doce sesiones en el lapso temporal de tres meses. Como muestra la Figura 4.1, los escenarios se categorizan como a) *controlled*, b) *degraded* y c) *adverse*. Esta base de datos consta de 52 usuarios (26 hombres y 26 mujeres) con 10 imágenes por sesión tal y como detalla la Tabla 4.1. Las imágenes fueron capturadas a partir de secuencias de vídeo mientras los usuarios leían un texto.

Individuos Totales	Hombres	Mujeres	Tamaño de las imágenes	Imágenes por individuo y sesión	Sesiones	Imágenes por sesión	Num. total imágenes
52	26	26	720x576	10	12	520	6240

Tabla 4.1: Datos sobre la base de datos BANCA.

Protocolo BANCA

La base de datos contiene una serie de protocolos para realizar experimentos donde se definen los conjuntos de entrenamiento y test necesarios para los experimentos de reconocimiento. En la Tabla 4.1 se muestra la información más relevante al contenido de la misma.

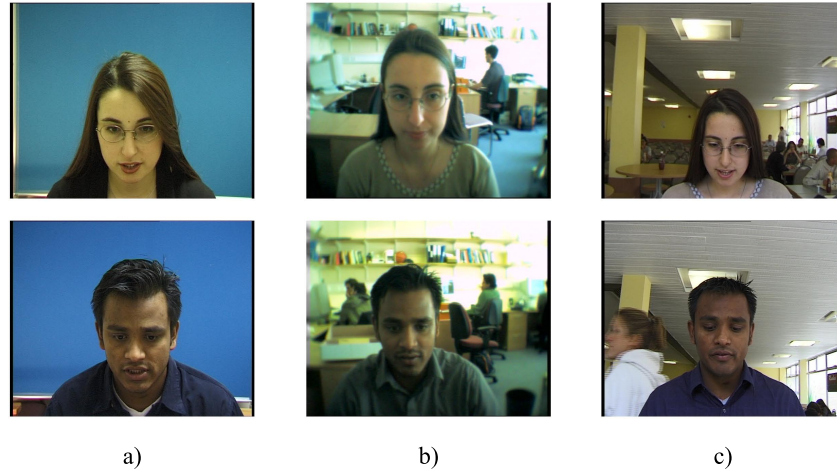


Figura 4.1: Muestra de imágenes en la base de datos BANCA. a) *controlled* b) *degraded* c) *adverse*

Para la base de datos de BANCA, el protocolo que se va a utilizar es el denominado como P en la documentación de la misma[3]. Para este protocolo, las imágenes de la primera sesión del grupo *controlled* serán las imágenes de entrenamiento y las imágenes de test serán las tres últimas sesiones de cada grupo. Hay que señalar que dentro de la base existen dos grupos llamados $g1$ y $g2$ dividiendo el total de usuarios en dos experimentos separados. La elección de este protocolo está basada en las similitudes que comparte con el sistema en tiempo real puesto que las imágenes entrenadas están relativamente controladas pero las imágenes que se tratan de reconocer pueden provenir de diferentes entornos.

4.1.2. Base de datos: FOCS

La base de datos de FOCS, está entablada en tres grandes grupos denominados a) *Good* b) *Bad* y c) *Ugly* (ver Figura 4.2). La separación en estos grupos no es, como en el caso anterior, por cambios en el entorno o sesión. Todas las imágenes tomadas fueron evaluadas y fueron separadas en función de los resultados de rendimiento que se obtuvieron con un sistema de referencia. Por tanto en el grupo *Good* se encuentran las imágenes que generalmente mejores resultados dan al usar el sistema de reconocimiento *baseline* descrito en [4], en el grupo *Bad* se encuentran las que dan malos resultados y, por último en el grupo *Ugly* se encuentran las que ofrecen peores resultados frente al reconocimiento [39].

Individuos totales	Hombres	Mujeres	Tamaño de las imágenes	Imágenes por individuo y grupo	Imágenes de entrenamiento	Imágenes de test	Num. Total imágenes
437	245	192	3008x2000	(Variable) 1-4	1085	1085	6510

Tabla 4.2: Datos sobre la base de datos FOCS.

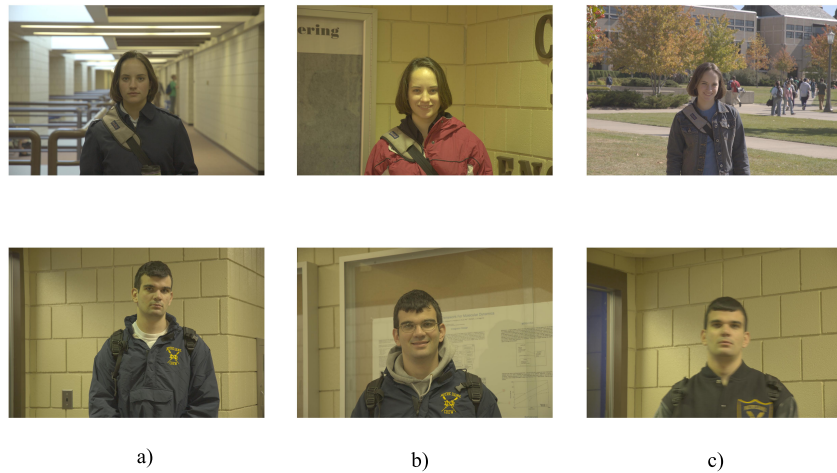


Figura 4.2: Muestra de imágenes en la base de datos FOCS. a) *good* b) *bad* c) *ugly*

Como se puede ver en la Tabla 4.2, la base de datos está compuesta por 437 usuarios (245 hombres y 192 mujeres). El entorno de las imágenes es variable y el número de imágenes que se toman por cada individuo también varía entre 1-4 imágenes haciendo un total de 1.085 imágenes por cada conjunto de entrenamiento y test. La distribución de imágenes por persona en ambos conjuntos (entrenamiento y test) es de 117 sujetos con 1 imagen, 122 sujetos con 2 imágenes, 68 sujetos con 3 imágenes y 130 sujetos con 4 imágenes Cabe resaltar que las imágenes tomadas para esta base de datos son de alta resolución si bien la distancia del individuo a la cámara es variable.

Protocolo FOCS

La base de datos de FOCS, a diferencia de BANCA, ya tiene las imágenes de cada grupo divididas en imágenes de entrenamiento y de test por lo que no es necesario realizar ninguna separación especial respecto a la misma. Tanto en entrenamiento como test, coinciden la cantidad de imágenes por usuario y el número total de imágenes en los conjuntos como muestra la Tabla 4.2. Los experimentos son separados en las tres categorías previamente descritas teniendo así conjuntos de entrenamiento y test para *good*, *bad* y *ugly* por separado.

4.2. Resultados experimentales

Para el presente proyecto se realizarán experimentos en tres niveles, detección, reconocimiento y tiempo de procesamiento analizando el sistema en cada una de estas tres etapas.

La siguiente sección se divide en las siguientes partes:

- **Rendimiento de las técnicas de detección (Exp.Det).** Los experimentos de detección se medirán a través del sistema off-line haciendo uso de las bases de datos presentadas. Los experimentos analizarán las fases de detección de cara (Exp.Det1) y detección (Exp.Det2) de ojos con las diferentes técnicas propuestas.

- **Rendimiento de las técnicas de reconocimiento (Exp.Rec).** Haciendo uso también del sistema off-line y de las bases de datos BANCA (Exp.Rec1) y FOCS (Exp.Rec2), se realizarán experimentos para evaluar todas las técnicas de reconocimiento implementadas y tener datos del rendimiento aproximado del sistema en tiempo real desarrollado.
- **Tiempos de procesamiento de cada etapa (Exp.Time).** Para las medidas de los tiempos de procesamiento, se hará uso del sistema en tiempo real. Se realizarán experimentos que contabilicen el tiempo que requiere cada etapa para ejecutarse y cómo afectarán diferentes factores como la resolución o la distancia de un usuario a las mismas.

4.2.1. Resultados de detección (Exp.Det)

Para medir el rendimiento de las técnicas de detección, se utilizan las bases de datos anteriormente expuestas en su totalidad. Para el caso particular de los ojos, su correcta detección está ligada a que haya una detección de cara correcta. Es por esto por lo que **sólo se utilizarán los subconjuntos de imágenes cuyas caras han sido correctamente extraídas por la anterior detección.**

Exp.Det1. Rendimiento de la detección de cara

En detección de cara, como se ha expuesto en el capítulo anterior, se utiliza el algoritmo de Viola-Jones. La ventana de detección menor se ha establecido en 20x20 píxeles. El sistema de detección es ejecutado sobre cada base de datos y de los datos extraídos se muestran la cantidad de fallos y el tipo de fallo que haya ocurrido (Figuras 4.4 y 4.5). Existirán dos tipos de fallos:

- **Fail To Acquire (FTA).** Fallo producido por no haber encontrado ninguna cara en la imagen.
- **Fail To Detect (FTD).** Fallo producido por haber detectado una zona donde no hay una cara.

A continuación se muestran los resultados para la base de datos BANCA:

Grupo	Número de imágenes	FTA	FTD	Usuarios con fallos	Total Fallos	Porcentaje de fallos
<i>Controlled</i>	2080	16	91	13	107	5,14%
<i>Degraded</i>	2080	55	31	21	76	3,61%
<i>Adverse</i>	2080	14	46	15	60	2,88%

Tabla 4.3: Resultados de la detección de cara en BANCA.

Adicionalmente a los datos mostrados en la Tabla 4.3, se incluyen unas gráficas con los fallos correspondientes a cada usuario (cuya detección haya previamente fallado). Esta información es interesante puesto que al haber bastantes imágenes por usuario (40 en cada grupo), se puede comprobar si existen muchos fallos agrupados en unos pocos usuarios o si, por el contrario los fallos se extienden a lo largo de la base de datos.

Como se puede comprobar en la Figura 4.3, el mayor número total de fallos se produce en el grupo *controlled* sin embargo, el número de usuarios que fallan son menos. Esto se debe a que

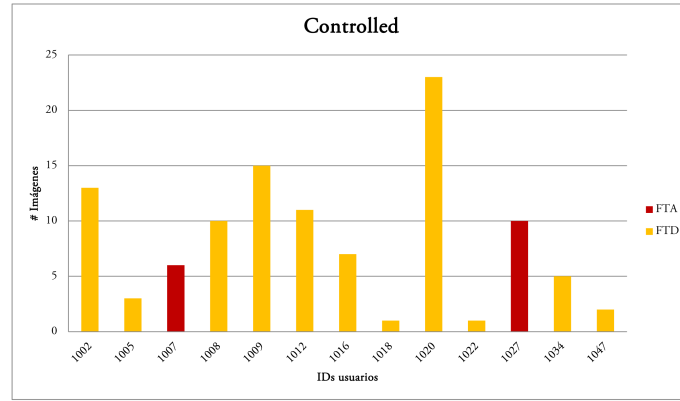
hay imágenes de usuarios en *controlled* cuya detección falla en toda una sesión haciendo que el número total de fallos se eleve rápidamente. Analizando los datos, se puede afirmar que el grupo *degraded* es el que peores resultados ofrece puesto que tiene el número más alto de usuarios cuya detección falla y el número total de fallos no es proporcionalmente bajo. Esto se debe a que las imágenes en este grupo son de menor calidad puesto que tienen filtros que degradan las mismas.

En el caso de FOCS, al tratarse de una base de datos con pocas imágenes del mismo usuario, la representación de las gráficas anteriores no otorga información relevante puesto que los experimentos devuelven detecciones fallidas con usuarios sin grandes cantidades de fallos acumulados. Una razón posible para que esto ocurra es el método de captura elegido para esta base de datos. Mientras que BANCA se basaba en la captura a partir de secuencias de vídeo, en FOCS las imágenes son fotografías con poca correlación entre ellas dentro del mismo usuario. Esto hace más propensa a los fallos en cadena a la base de datos de BANCA que a la de FOCS.

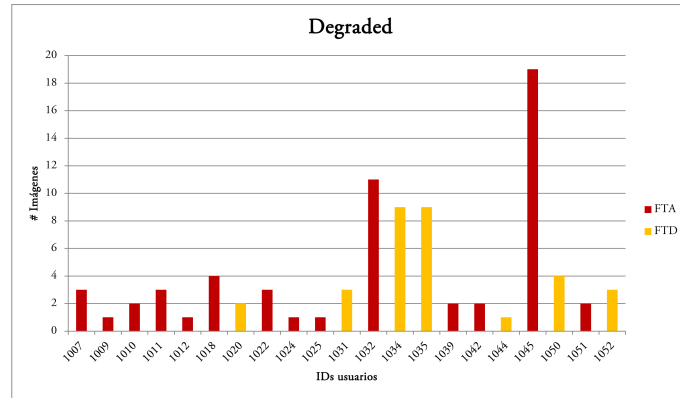
Grupo	Núm. imágenes	FTD	Porcentaje de fallos
<i>Good</i>	2170	90	4.15%
<i>Bad</i>	2170	145	6.68%
<i>Ugly</i>	2170	199	9.17%

Tabla 4.4: Resultados de la detección de cara en FOCS.

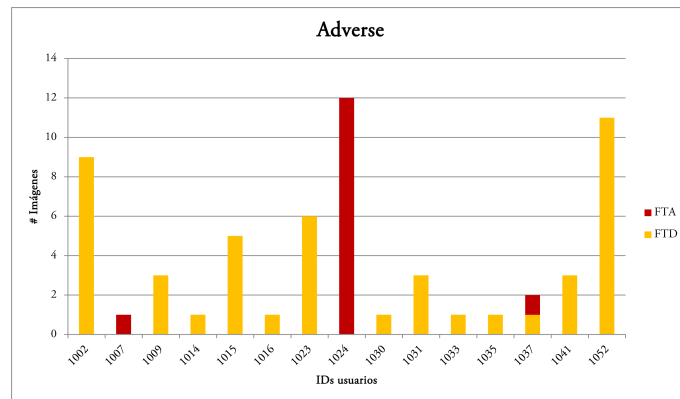
En la Tabla 4.4 se muestran los resultados relevantes a la detección para la base de datos FOCS. Dicha tabla refleja como empeoran los resultados consecuentemente al grupo estudiado. Sólo se representan los datos para detecciones incorrectas (FTD) puesto que durante los experimentos no hubo ningún caso en el que no se detectara nada (FTA=0). Esto es debido a que las imágenes de FOCS son de gran resolución (3008x2000 píxeles) por lo que las probabilidades de que el algoritmo no de una posible detección se reducen ya que la ventana más pequeña para la detección es de 40x40 píxeles. Con imágenes tan grandes hay muchas posibilidades de tener una coincidencia con cualquier objeto del fondo, es decir, gran cantidad de falsas detecciones.



a)



b)



c)

Figura 4.3: Usuarios con fallos en la detección de cara en los grupos a) *controlled*, b) *degraded* y *adverse*

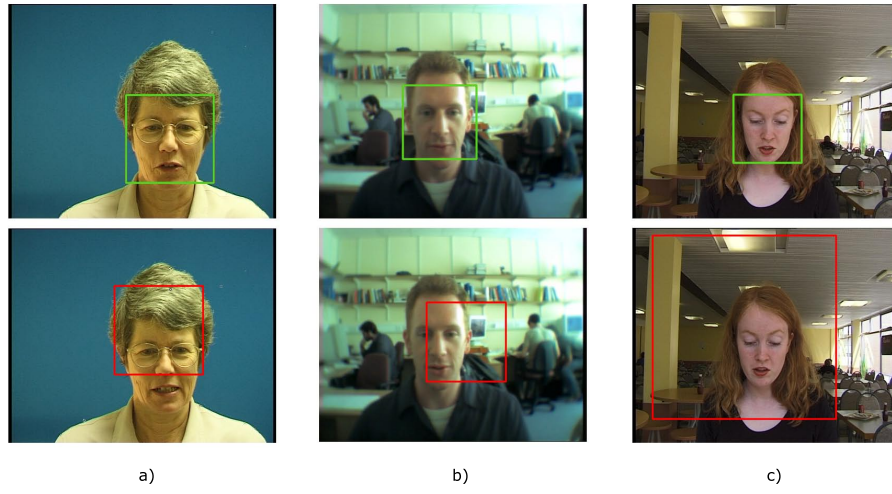


Figura 4.4: Ejemplos de detección correcta e incorrecta para a) *Controlled* b) *Degraded* c) *Adverse*.

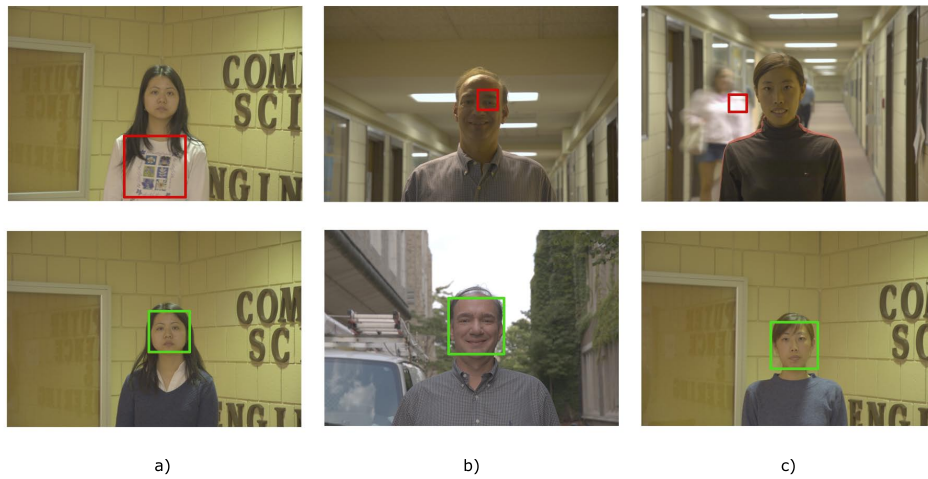


Figura 4.5: Ejemplos de detección correcta e incorrecta para a) *Good* b) *Bad* c) *Ugly*.

Exp.Det2. Rendimiento de la detección de ojos

Para los experimentos en la detección de ojos, como se ha comentado anteriormente, se utilizarán las imágenes de las bases de datos cuya detección de cara no falló ($A = Total - FTD - FTA$). Se han utilizado dos técnicas de detección de ojos, i) mediante el algoritmo de Viola-Jones y ii) mediante el método de encontrar los picos de los histogramas de las imágenes binarizadas. En estos experimentos, se comparará cada técnica y se contabilizarán los fallos para estos casos:

- Fallo en la detección del ojo derecho (FTA_{RE}).
- Fallo en la detección del ojo izquierdo (FTA_{LE}).
- Fallo en la detección de ambos ojos (FTA_{BE}).

Los fallos en ambos ojos no son contabilizados dentro de los fallos de cada ojo ($FTA_{Total} = FTA_{RE} + FTA_{LE} + FTA_{BE}$). Debido al reducido área de la imagen, los fallos correspondientes al algoritmo de Viola-Jones siempre vienen dados por no detectar nada en la imagen (FTA). Los fallos producidos por el método de imágenes binarizadas, son debido a una estimación incorrecta de la localización del ojo.

En la Tabla 4.5 se muestran los resultados de la detección para la base de datos BANCA.

Técnica	Grupo	Imágenes (A)	FTA_{RE}	FTA_{LE}	FTA_{BE}	FTA_{Total}	Porcentaje de fallos
Histogramas de la imagen binarizada	<i>Controlled</i>	1973	132	141	174	447	22,66%
	<i>Degraded</i>	2004	59	94	99	252	12,57%
	<i>Adverse</i>	2020	86	34	92	212	10,49%
Algoritmo de Viola-Jones	<i>Controlled</i>	1973	96	26	126	248	12,57%
	<i>Degraded</i>	2004	21	322	58	401	20,00%
	<i>Adverse</i>	2020	126	162	169	457	22,62%

Tabla 4.5: Resultados de la detección de ojos en BANCA.

Debido a la procedencia de las imágenes (secuencias de vídeo de calidad media) y a que la mayoría de los usuarios aparecen mirando hacia abajo, la tasa de errores en detección de ojos es relativamente alta. Dado que el elemento a detectar es pequeño, uno de los factores que más influyen en el algoritmo de Viola-Jones es la calidad de la imagen y, como se puede comprobar, este método es el que peores resultados obtiene destacando los fallos de ojos izquierdos en el grupo *degraded*. Esta cantidad tan alta de fallos puede resultar de los filtros de degradación unidos un entorno especialmente desfavorable para el algoritmo (la localización no cambia por lo que elementos como la iluminación serán similares en toda la base de datos).

En el caso de la base de datos FOCS, las imágenes tomadas son de gran calidad por lo que se espera que la detección mediante el algoritmo de Viola-Jones sea más efectiva. En la Tabla 4.6 se muestran los resultados relevantes a la detección de ojos en la base de datos FOCS. Como se puede ver en este caso, el algoritmo de Viola-Jones es claramente superior teniendo valores siempre por debajo de los ofrecidos por el método de binarización. Se pone de manifiesto la importancia de la resolución de las imágenes para el algoritmo de Viola-Jones.

Hay que resaltar que, independientemente de la base de datos utilizada, cuando existe un acierto el algoritmo de Viola-Jones obtiene una precisión de localización muy superior al método

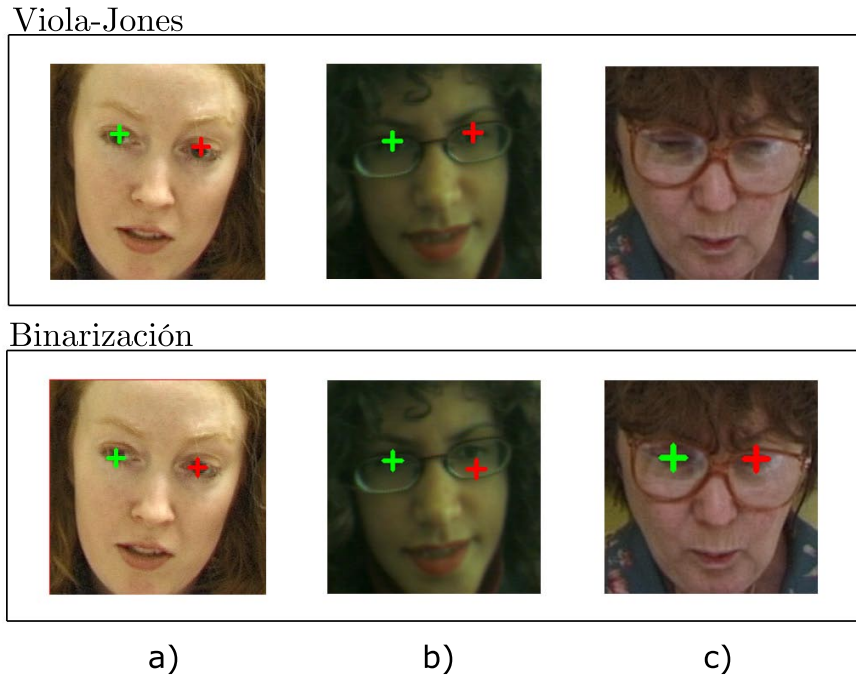


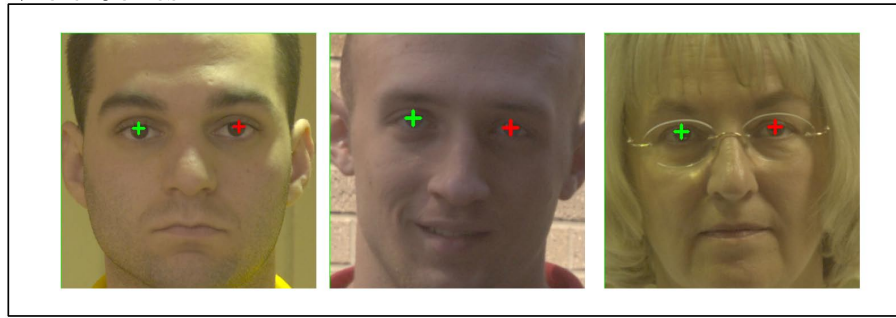
Figura 4.6: Comparación de técnicas en a) *Controlled* b) *Degraded* c) *Adverse*.

Técnica	Grupo	Imágenes (A)	FTA _{RE}	FTA _{LE}	FTA _{BE}	FTA _{Total}	Porcentaje de fallos
Histogramas de la imagen binarizada	<i>Good</i>	2080	28	49	174	251	12,06%
	<i>Bad</i>	2025	39	44	99	182	8,99%
	<i>Ugly</i>	1971	37	41	92	170	8,62%
Algoritmo de Viola-Jones	<i>Good</i>	2080	5	8	19	32	1,54%
	<i>Bad</i>	2025	14	29	20	63	3,11%
	<i>Ugly</i>	1971	32	84	38	154	7,81%

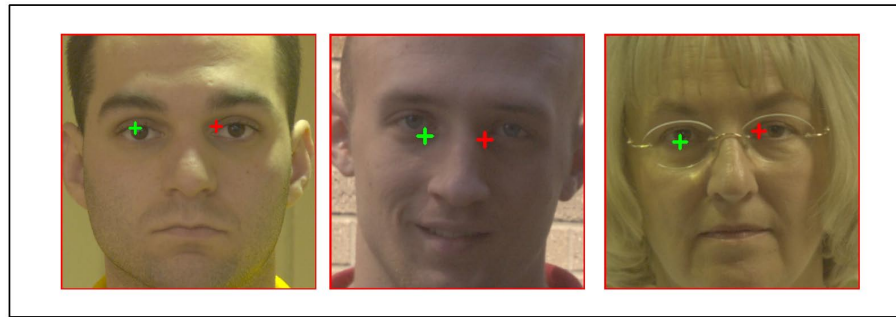
Tabla 4.6: Resultados de la detección de ojos en FOCS.

de binarización tal y como se puede ver en las Figuras 4.6 y 4.7. La precisión durante la detección de ojos es crítica si el sistema depende de unas imágenes lo mejor normalizadas posible para su correcto funcionamiento como es el caso del sistema desarrollado en este proyecto.

Viola-Jones



Binarización



a)

b)

c)

Figura 4.7: Comparación de técnicas en a) *Good* b) *Bad* c) *Ugly*.

4.2.2. Resultados de reconocimiento (Exp.Rec)

Para medir el rendimiento de las técnicas de reconocimiento utilizadas, se llevan a cabo los experimentos tal cual marcan los protocolos de las bases de datos detallados al principio de este capítulo. Los resultados se mostrarán en forma de curvas Receiver Operating Characteristic (ROC) junto con el Equal Error Rate (EER), punto de trabajo del sistema donde se obtiene $FA = FR[40]$. Las técnicas y sus respectivos parámetros utilizados en los experimentos son los siguientes:

- **System1. PCA+SVM.** Para la reducción de características con PCA, se han escogido 200 componentes principales puesto que con esta cantidad la pérdida de información es mínima. En relación a los SVM, se entrenan con la siguiente configuración de parámetros:
 - Kernel lineal
 - $\gamma = 20$
 - Coste, $C = 1000$
 - Sin pesos por clase
 - Iteraciones máximas; 1000
 - Error, $\epsilon = 1e - 6$

En este caso, los resultados se ven altamente afectados por el tipo de kernel que utilice el SVM. El kernel lineal es el que peores resultados ofrece frente a otros como el gaussiano pero a su vez es el más rápido. En este proyecto se ha optado por utilizar el kernel lineal

y asumir tasas de reconocimiento menores puesto que los tiempos de procesamiento de kernels más complejos son menos viables en tiempo real.

- **System2. LBP+Chi-Square.** En el caso de la técnica de LBP con Chi-Square, los parámetros utilizados son los que, según las publicaciones al respecto, suelen dar resultados óptimos [24]. Dichos parámetros son:

- Operador extendido de radio $R = 2$
- Número de muestras $P = 8$
- División de la imagen en $D = 7 \times 7$ regiones

Debido a que la técnica de LBPs obtiene resultados diferentes si se modifican estos parámetros, se procedió a aumentar el radio del vecindario ($R = 8$) y a disminuir el tamaño de las divisiones ($D = 10 \times 10$) con el objetivo de mejorar el rendimiento del mismo. Este sistema se denominará System2mod refiriéndose al sistema LBP con parámetros ajustados.

- **System3. Fusión.** La fusión que se realiza de las técnicas de PCA y LBP es una fusión a nivel de score. Para que los resultados obtenidos se puedan fusionar, se deben normalizar. El tipo de normalización elegido es la normalización por tangente hiperbólica debido a los buenos resultados que ofrece respecto a otras técnicas[37].

Exp.Rec1. Resultados sobre la base de datos BANCA

A continuación se muestran en la Figura 4.8 los resultados de los experimentos sobre la base de datos BANCA para el protocolo P y los grupos $g1$ y $g2$:

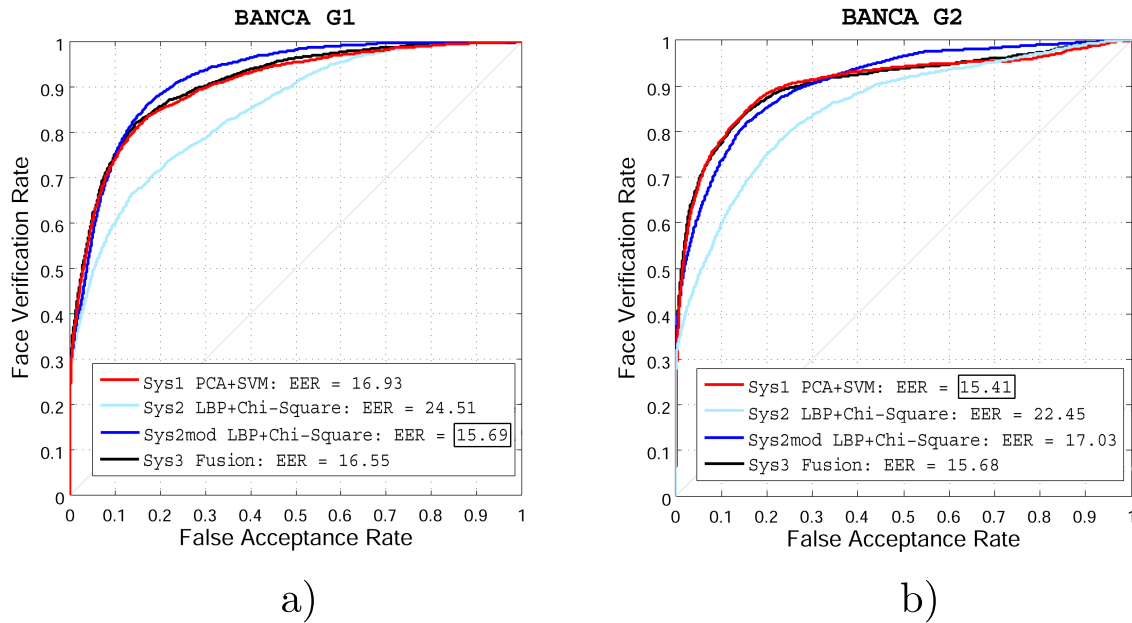


Figura 4.8: Curva ROC para las técnicas implementadas en BANCA a) grupo $g1$ b) grupo $g2$.

Como se puede comprobar, la técnica de LBP con los parámetros ajustados, mejora sensiblemente los resultados obtenidos respecto a la técnica con los parámetros preestablecidos.

Los resultados de PCA con SVM obtienen resultados parecidos a los LBPs con los parámetros modificados, siendo mejores en el grupo *g2*. En la gráfica también se puede comprobar que la fusión entre ambas técnicas no llega a mejorar el sistema de forma significativa. En la Tabla 4.7 se muestran algunos resultados obtenidos con otras técnicas LBP en la base de datos BANCA junto con los resultados obtenidos de los sistemas implementados. El sistema *LBP Ahonen* [24] es el **más similar al desarrollado en este proyecto**.

Models	BANCA	
	P	
LDA/NC	15.5	
DCTmod2/GMM	18.6	
LBP Ahonen	20.8	
INORM LBP/HMM	11.7	
LBP/MAP	19.2	
	g1	g2
Sys1 PCA/SVM	16.6	15.4
Sys2 LBP/ χ^2	24.3	22.4
Sys2mod LBP/ χ^2	15.7	17.0
Sys3 FUSION	16.9	15.6

Tabla 4.7: Comparación de tasas de error en varios sistemas que utilizan LBP (junto con dos métodos del estado del arte) y los implementados. Tabla extraída de [1].

Los datos extraídos de la Tabla 4.7 muestran cómo para cada grupo hay un sistema que obtiene los mejores resultados quedando todos menos *system2* por encima de algunos de los sistemas de referencia. También se puede observar que la fusión no llega a mejorar las tasas de error por lo que se puede decir que no son dos sistemas que ofrezcan buenos resultados fusionando a nivel de score.

Exp.Rec2. Resultados sobre la base de datos FOCS

Para los experimentos con la base de datos de FOCS, se han utilizado los mismos parámetros para las técnicas de reconocimiento que en el caso de BANCA. En la Figura 4.9, se muestran los resultados de las diferentes técnicas en cada uno de los grupos. La Figura 4.10 muestra los resultados del mejor sistema (system2mod) en el punto de FAR=0.001 con respecto al sistema *baseline*.

Como muestra la Figura 4.9, PCA es superior a las técnicas por LBP excepto en el grupo *ugly* donde los LBPs obtienen mejores resultados y además contemplan la mayor diferencia frente a LBP sin ajustar parámetros.

El sistema *baseline* utiliza técnicas de PCA por lo que sirve de punto de comparación con el PCA desarrollado en el proyecto. En dicho sistema y para minimizar los efectos de variaciones de luminosidad, se aplica un método conocido como *self-quotient image* que se basa en la creación de una segunda imagen a partir de un filtrado gaussiano para después dividir la imagen original por la filtrada elemento a elemento. Esto crea una imagen resultante que ofrece poca variación a los cambios de iluminación. Adicionalmente, en la imagen se realizan catorce proyecciones diferentes de PCA localmente para crear un vector de características extendido. Este procedimiento, aunque pueda producir mejores resultados, tiene una carga computacional elevada para una aplicación en tiempo real.

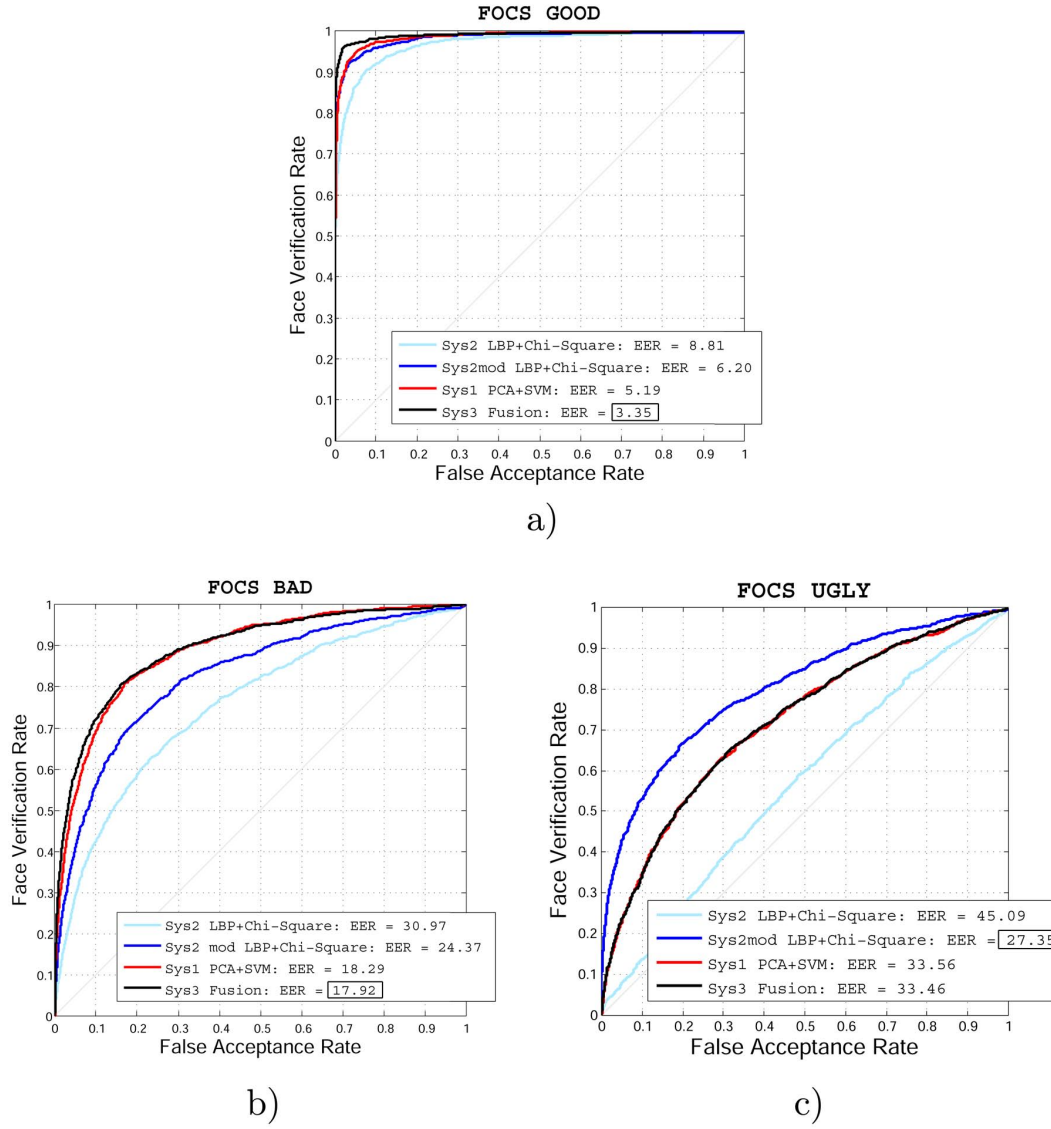


Figura 4.9: Curvas ROC para las técnicas implementadas en los grupos a) *good* b) *bad* y c) *ugly* de FOCS.

Respecto a la Figura 4.10, se comprueba como el sistema desarrollado obtiene mejores resultados que el de referencia en *good* y *ugly* sin embargo en *bad* no se obtienen tan buenos resultados. Estos datos pueden deberse a la naturaleza de la base de datos ya que en *bad* se agrupan el mayor número de imágenes tomadas en el exterior cuyas características pueden hacer que el sistema no funcione tan bien.

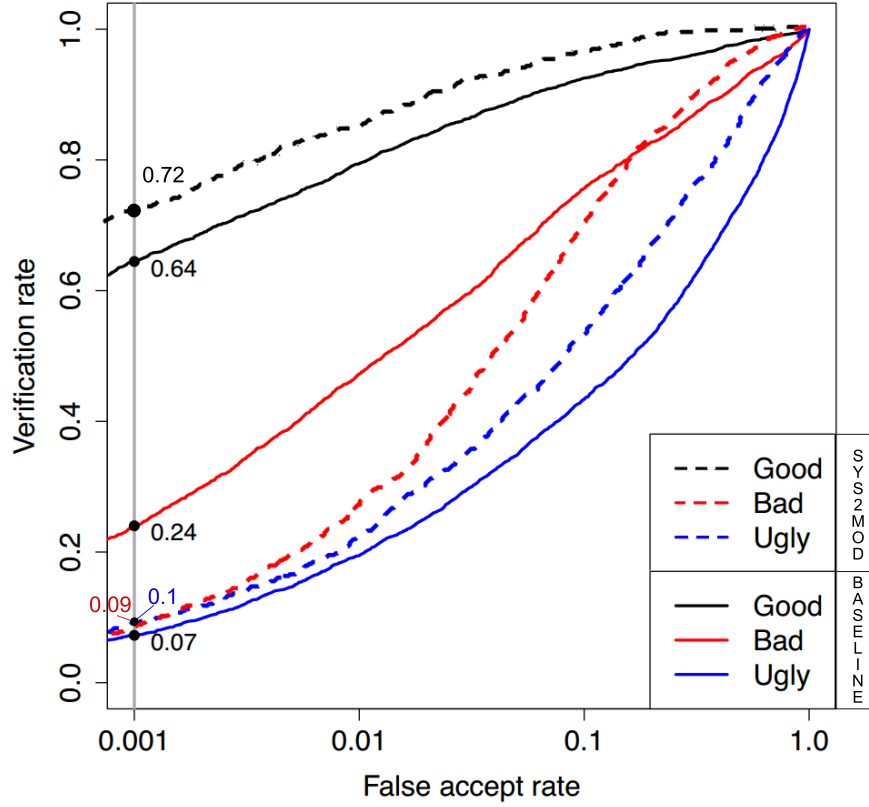


Figura 4.10: Comparación de los resultados del sistema *baseline* respecto a los obtenidos con el sistema *system2mod*.

4.2.3. Resultados de los tiempos de procesamiento de las etapas (Exp.Time)

El objetivo de este último experimento es el de medir los tiempos de procesamiento que requiere cada etapa del sistema en tiempo real en modo identificación (Exp.Time1), demostración (Exp.Time.2) y entrenamiento (Exp.Time3) para evaluar en que magnitud afectan ciertos factores como cambios en la resolución o la distancia de un individuo a la cámara.

Es por ello que, para cada modo de funcionamiento, se contabilizarán los tiempos medios de cada etapa involucrada modificando la resolución en el caso de identificación y la distancia en el caso de demostración. Para obtener los tiempos, se utiliza un temporizador de Qt que utiliza el reloj interno del sistema [6]. Es importante comentar que, debido al funcionamiento del sistema operativo, **el reloj tiene un error de $\pm 5ms$** . Todos los resultados son obtenidos mediante el equipamiento definido en esta memoria y están, por tanto, fuertemente ligados a las características técnicas del mismo (frecuencia del procesador, velocidad de lectura/escritura de los discos, etc.).

Exp.Time1. Modo identificación

En el modo identificación, los tiempos a estudiar son; t_{dc} , t_{do} , t_{pp} , t_{ex} y t_m . Estas variables corresponden al tiempo que tarda en ejecutarse cada etapa involucrada en el sistema. En la Figura 4.11 se puede ver a que etapa del sistema corresponde cada uno.

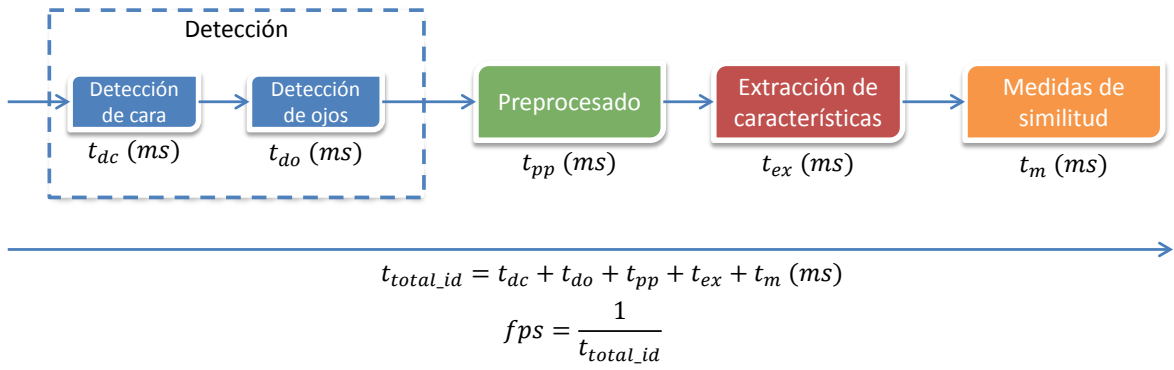


Figura 4.11: Diagrama de etapas del sistema en tiempo real de las cuales se extraen los tiempos de procesamiento.

La distancia se fijará en 1 metro puesto que esta afectará a la etapa de detección que será estudiada en el siguiente experimento. En la técnica de fusión, los tiempos de extracción de características y medida de similitud de son almacenados en una única variable. La base de datos tiene un total de 200 imágenes durante la realización de los experimentos

Técnica	Resolución	t_{dc}	t_{do}	t_{pp}	t_{ex}	t_m	t_{total_id}	Fotogramas por segundo
LBP+Chi-Square	640x480	59	3	32	16	35	145	6,9
	1280x720	95	7	74	18	34	228	4,4
	1920x1080	169	28	136	22	41	396	2,5
LBP+Chi-Square (Parámetros modificados)	640x480	32	5	29	17	219	302	3,31
	1280x720	73	12	56	16	213	370	2,7
	1920x1080	120	27	117	17	224	505	2
PCA+Dist. Euclídea	640x480	45	4	37	14	2	102	9,8
	1280x720	72	11	64	13	2	162	6,2
	1920x1080	128	27	118	15	2	290	3,4
Fusión	640x480	52	3	34	63		152	6,6
	1280x720	91	7	74	64		236	4,2
	1920x1080	142	23	141	64		370	2,7

Tabla 4.8: Medidas de tiempos (ms) para cada etapa del sistema (los resultados pueden tener un error de $\pm 5ms$).

Como se puede observar en la Tabla 4.8, las etapas de detección y procesado se ven altamente influenciadas por la resolución seleccionada mientras que las etapas de extracción de características y medidas de similitud permanecen relativamente constantes. Estas medidas aumentarán su tiempo de procesado conforme la base de datos se amplíe. Como es de esperar, los tiempos de procesado de distancia euclídea son muy bajos puesto que el tamaño de los vectores de características producidos por PCA (dimensión: 200) son mucho menores que los producidos por los LBPs (dimensión: 2891) y la versión con modificación de parámetros (dimensión: 5900).

Exp.Time2. Modo demostración

En el modo demostración sólo entra en juego la detección de cara. Según la implementación de la función de OpenCV para detectar objetos, la detección puede realizarse de sólo un objeto o de múltiples objetos. Con este experimento se pretende estudiar el efecto que tiene la distancia

del objeto a detectar en el tiempo de procesado. Para realizar dicho experimento, se realizan medidas consecutivas del tiempo de detección de una cara (t_{dc}) desde una distancia de 5 metros hasta 0,5 metros de la cámara en las tres diferentes resoluciones descritas en el capítulo anterior.

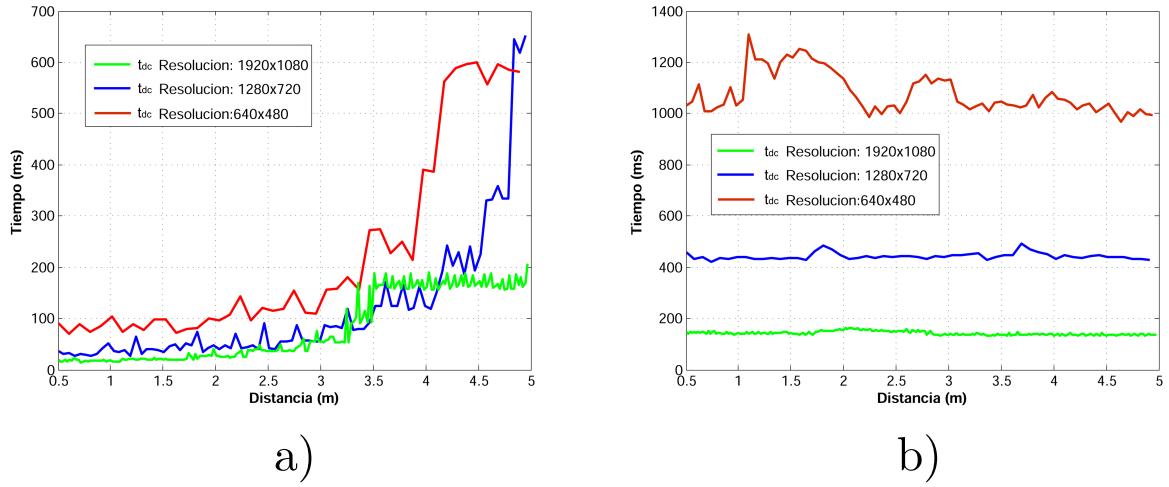


Figura 4.12: Tiempos de proceso para a) la detección de una cara y b) detección múltiple.

Como se puede observar en la Figura 4.12, para el caso de un sólo objeto a detectar, el tiempo de procesado aumenta conforme el objeto se aleja de la cámara. Esto es debido a que en el momento en que el algoritmo encuentra un objeto, deja de realizar barridos en la imagen por lo que cuanto mayor sea el objeto detectado menos tardará ya que se ahorra todo el tiempo que sería invertido en proseguir con el barrido. En la detección múltiple, sin embargo, los tiempos permanecen relativamente constantes teniendo una mayor fluctuación cuanto más alta es la resolución.

Los tiempos mínimos de detección son conseguidos por la resolución más baja en detección de una cara y oscilan en el rango de 25 ms. Obteniendo los fotogramas por segundo resulta en $\frac{1}{0,025} = 40fps$ por lo que en este caso el sistema funcionaría a la velocidad máxima de la cámara (30 fps). Los tiempos máximos en detección de una cara oscilan en los 600 ms, lo que se traduce en 1,67 fotogramas por segundo. Teniendo en cuenta que el ojo humano empieza a notar discontinuidades por debajo de 15 fps, los picos de procesado producirán una gran ralentización haciendo que el vídeo se vea a saltos.

Exp.Time3. Modo entrenamiento

Para el modo entrenamiento, se han realizado medidas de las etapas involucradas, detección (t_{dc} y t_{do}) y preprocesado (t_{pp}) (ver Figura 4.13) con el objetivo de comprobar la velocidad de imágenes por segundo que es capaz de obtener, es decir, la velocidad del entrenamiento.

En la Tabla 4.9 se muestran los resultados obtenidos para cada tipo de resolución. El coste por entrenar con una calidad de imagen mayor es un tiempo más elevado para capturar un número determinado de imágenes, algo que puede ser crítico según la aplicación del sistema.

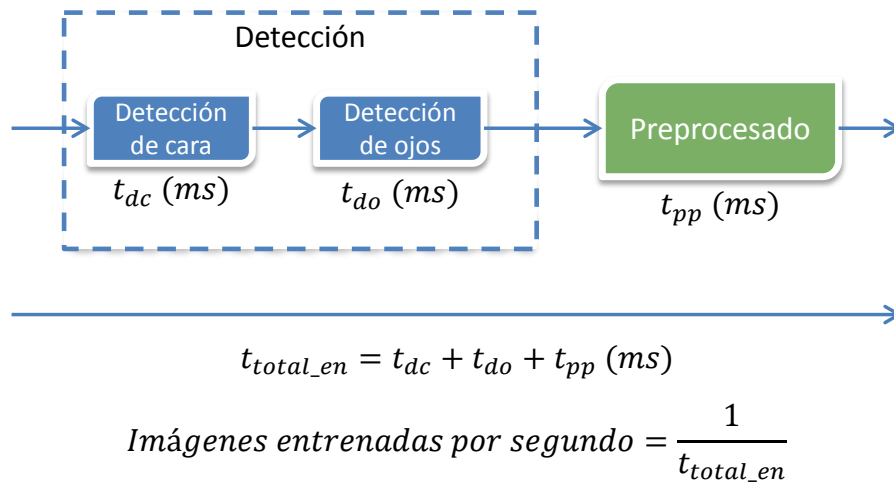


Figura 4.13: Diagrama de tiempos para las etapas involucradas en el modo entrenamiento.

Resolución	t_{dc}	t_{do}	t_{pp}	t_{total_en}	Velocidad de entrenamiento (imágenes por segundo)
640x480	40	3	34	77	13
1280x720	88	8	71	167	6
1920x1080	157	26	149	332	3

Tabla 4.9: Medidas de tiempos (ms) para cada etapa del sistema en modo entrenamiento.

5

Conclusiones y trabajo futuro

5.1. Conclusiones.

En el presente proyecto se han estudiado, desarrollado, implementado, evaluado y documentado dos sistemas completos de reconocimiento facial, uno off-line para el trabajo con bases de datos del estado del arte y otro para el funcionamiento en tiempo real. Dichos sistemas han sido evaluados en diferentes aspectos como el rendimiento en detección y reconocimiento y como el tiempo de procesamiento de cada etapa (aspectos relevantes en el reconocimiento en tiempo real).

En primer lugar, para el funcionamiento de los sistemas se han utilizado técnicas del estado del arte para cada etapa, implementándose dos técnicas diferentes en la detección y dos técnicas, una basada en características globales y otra en características locales, para el reconocimiento. La utilización de diferentes técnicas permite su estudio y comparación frente a diferentes escenarios impuestos por las bases de datos elegidas. Tras el desarrollo completo de los dos sistemas, se han llevado a cabo experimentos que tienen como objetivo evaluar las etapas con mayor influencia sobre el funcionamiento global. Gracias a la posibilidad de generar resultados a partir de las bases de datos por parte del sistema off-line, ha sido posible medir la precisión tanto de la etapa de detección como de la etapa de reconocimiento y así obtener una precisión aproximada para el sistema on-line. En el caso del sistema on-line, uno de los aspectos más importantes es el tiempo que se tarda en el proceso de las etapas para garantizar un funcionamiento en tiempo real.

Las dos bases de datos utilizadas en los experimentos, permiten tener información sobre el rendimiento en diferentes condiciones. Por una parte, la base de datos de BANCA otorga imágenes de secuencias de vídeo de una calidad media como podrían obtenerse a partir de un sistema de videovigilancia. Por otro lado, la base de datos de FOCS proporciona imágenes de alta calidad con una gran cantidad de usuarios únicos y con imágenes de entrenamiento variables asemejándose a lo que podría ser la base de datos de una comisaría.

En un sistema de reconocimiento facial del que no se dispone de información adicional como las coordenadas de los ojos, la detección automática juega un papel crucial ya que de realizarse incorrectamente, producirá casi seguramente un fallo en el resto de etapas siguientes. Los experimentos realizados referentes a la detección de cara, muestran una buena precisión con tasas de fallo bajas. Se puede comprobar que la tasa de fallos aumenta cuanto más grande sea la imagen

puesto que hay más probabilidades de detectar erróneamente una cara. También se ponen de manifiesto los fallos en cadena que suelen aparecer en secuencias de vídeo. Para la detección de ojos, de las dos técnicas implementadas, Viola-Jones tiene una tasa de fallos mayor que en detección de cara debido a que la imagen a detectar en este caso suele ser de pequeño tamaño y tanto la resolución como oclusiones por sombras pueden hacer que la técnica falle. La técnica de binarización obtiene menos fallos pero generalmente es menos precisa al localizar el centro del ojo. Es por ello que en una aplicación en tiempo real y aprovechando el hecho de trabajar con secuencias de vídeo, la técnica de Viola-Jones descartará un mayor número de fotogramas pero los que se obtengan por una detección válida tendrán mucha mayor precisión facilitando el mejor funcionamiento del resto de etapas.

Las etapas de extracción de características y medidas de similitud conforman uno de los principales módulos relativos tanto al rendimiento en términos de tasa de reconocimiento como de velocidad de procesamiento. Como el objetivo es que el sistema sea funcional en tiempo real, se eligieron técnicas de bajo coste computacional y alta reducción de la dimensionalidad a la vez que sean capaces de dar buenos resultados en reconocimiento. Con vectores de características de baja dimensionalidad se consigue que la etapa de medida de similitud se realice en menor tiempo al tener que realizar una menor cantidad de operaciones. Evaluando los resultados obtenidos respecto a la identificación se puede concluir que, si bien el objetivo principal del proyecto no era el de mejorar las técnicas ya existentes, los resultados obtenidos están en ocasiones por encima de los conseguidos por otras técnicas con, a priori, mayor coste computacional. Mediante dichos experimentos también se pone de manifiesto la importancia de ajustar los parámetros de la técnica de LBP puesto que **se producen mejoras relativas de hasta un 60,5 % en las EER** (caso FOCS *ugly*).

Por último, los experimentos relativos a los tiempos de procesamiento revelan que, aunque no sea posible llegar a funcionar al máximo de velocidad permitido por la cámara, el sistema alcanza unas tasas de fotogramas por segundo razonablemente altas para poder ser usadas en tiempo real. El cambio de resolución es uno de los factores que más influyen en los tiempos aunque se debe tener en cuenta que las mayores resoluciones permiten detecciones a más distancia (imágenes más grandes) y, generalmente, dan mejores resultados puesto que en imágenes de cara pequeñas no se degradan en tanta medida como las imágenes de poca resolución.

5.2. Trabajo futuro

Con el objetivo de continuar y mejorar lo estudiado en este proyecto, se proponen diferentes líneas de trabajo:

- Desarrollo y evaluación de nuevas técnicas que permitan el funcionamiento en tiempo real. Entre otras, la variante de PCA, LDA también conocida como *fisherfaces* es una técnica de similar coste computacional con la que se obtiene mayor robustez frente a cambios de iluminación. Una de las técnicas que mejores resultados están dando en la actualidad son las que usan DCT y GMM obteniendo además unos tiempos de proceso muy bajos.
- Implementación de paralelismo en las etapas del sistema. Mediante la ejecución de las etapas en múltiples procesadores se puede mejorar drásticamente la velocidad del sistema pudiéndose conseguir tasas de fotogramas por segundo máximas lo que convierten a esta línea de trabajo en una de las más interesantes.

- Adaptación del sistema para el uso de cámaras IP. La posibilidad de utilizar cámaras IP permiten que el sistema pueda ser controlado de forma remota mientras que el escenario de estudio se encuentra localizado en otro lugar.
- Detección y reconocimiento de múltiples caras. Una de las funciones que se beneficiarían directamente de la implementación de paralelismo en el sistema es la de el funcionamiento para múltiples caras en la misma imagen ya que utilizando un proceso para cada cara, los tiempos no serían muy superiores al funcionamiento con una cara.
- Evaluación de diferentes bases de datos adicionales. Mediante el uso de otras bases de datos que estén entabladas en varios entornos de captura, progresivamente se obtiene una información sobre el comportamiento del sistema más cercana a la realidad.
- Desarrollo de una decisión por umbral para permitir un funcionamiento del sistema en conjunto abierto.

Glosario de acrónimos

- **API**: Application Programming Interface
- **DCT**: Discrete Cosine Transform
- **DET**: Detection Error Tradeoff
- **DFT**: Discrete Fourier Transform
- **DWT**: Discrete Wavelet Transform
- **EBGM**: Elastic Bunch Graph Matching
- **EER**: Equal Error Rate
- **FA**: False Acceptance
- **FAR**: False Acceptance Rate
- **FOCS**: Face and Ocular Challenge Series
- **FR**: False Rejection
- **FRR**: False Rejection Rate
- **FTA**: Fail To Acquire
- **FTD**: Fail To Detect
- **GMM**: Gaussian Mixture Models
- **HMM**: Hidden Markov Models
- **IPD**: Interpupillary Distance
- **KNN**: K-Nearest Neighbours
- **LBP**: Local Binary Patterns
- **LDA**: Linear Discriminant Analysis
- **NIST**: National Institute of Technology
- **PCA**: Principal Component Analysis
- **ROC**: Receiver Operating Characteristic
- **SDK**: Software Development Kit
- **SVM**: Support Vector Machine

Bibliografía

- [1] Sébastien Marcel, Yann Rodriguez, and Guillaume Heusch. On the recent use of local binary patterns for face authentication. *International Journal on Image and Video Processing Special Issue on Facial Image Processing*, 0 2007. IDIAP-RR 06-34, accepted for publication but withdrawn because of author charges.
- [2] Ajay Kumar Anil K. Jain. Biometrics of next generation: An overview. In *Second Generation Biometrics*. Springer, 2010.
- [3] Enrique Bailly-baillire, Samy Bengio, Frédéric Bimbot, Miroslav Hamouz, Josef Kittler, Johnny Mariéthoz, Jiri Matas, Kieron Messer, Fabienne Porée, and Belen Ruiz. The banca database and evaluation protocol. In *In Proc. Int. Conf. on Audio- and Video-Based Biometric Person Authentication (AVBPA03)*, pages 625–638. Springer-Verlag, 2003.
- [4] P. Jonathon Phillips, J. Ross Beveridge, Bruce A. Draper, Geof H. Givens, Alice J. O’Toole, David S. Bolme, Joseph P. Dunlop, Yui Man Lui, Hassan Sahibzada, and Samuel Weimer. An introduction to the good, the bad, & the ugly face recognition challenge problem. In *FG*, pages 346–353. IEEE, 2011.
- [5] Opencv documentation page. <http://docs.opencv.org/>.
- [6] Qt documentation page. <http://qt-project.org/doc/qt-5.0/qt5doc/index.html>.
- [7] Woodrow Wilson Bledsoe. The model method in facial recognition. Technical report, Panoramic Research Inc, 1966.
- [8] T. Kohonen. *Self-organization and associative memory: 3rd edition*. Springer-Verlag New York, Inc., New York, NY, USA, 1989.
- [9] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of The Optical Society of America A-optics Image Science and Vision*, 4, 1987.
- [10] M.A. Turk and A.P. Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR ’91., IEEE Computer Society Conference on*, pages 586–591, 1991.
- [11] Anil K. Jain and Stan Z. Li. *Handbook of Face Recognition*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.
- [12] Zhifeng Li, Unsang Park, and Anil K. Jain. A discriminative model for age invariant face recognition. *IEEE Transactions on Information Forensics and Security*, 6(3-2):1028–1037, 2011.

- [13] Conrad Sanderson and Brian C. Lovell. Multi-region probabilistic histograms for robust and scalable identity inference. In *Proceedings of the Third International Conference on Advances in Biometrics*, ICB '09, pages 199–208, Berlin, Heidelberg, 2009. Springer-Verlag.
- [14] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1, 2001.
- [15] Phillip Ian Wilson and Dr. John Fernandez. Facial feature detection using haar classifiers. *Journal of Computing Sciences in Colleges*, pages 127–133, 2006.
- [16] A. Nabatchian, E. Abdel-Raheem, and M. Ahmadi. Human face recognition using different moment invariants: A comparative study. In *Image and Signal Processing, 2008. CISP '08. Congress on*, volume 3, pages 661–666, 2008.
- [17] John C. Russ. *Image Processing Handbook, Fourth Edition*. CRC Press, Inc., Boca Raton, FL, USA, 4th edition, 2002.
- [18] ISO/IEC 19794-5:2011. Information technology – biometric data interchange formats – part 5: Face image data. In *International Organization for Standardization*, 2011.
- [19] I.T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 1986.
- [20] P.N. Belhumeur, J.P. Hespanha, and D. Kriegman. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):711–720, 1997.
- [21] Ziad M. Hafed and Martin D. Levine. Face recognition using the discrete cosine transform. *Int. J. Comput. Vision*, 43(3):167–188, July 2001.
- [22] Ingemar J. Cox, J. Ghosn, and P.N. Yianilos. Feature-based face recognition using mixture-distance. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, pages 209–216, 1996.
- [23] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(12):2037–2041, December 2006.
- [24] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face recognition with local binary patterns. In Tomás Pajdla and Jiří Matas, editors, *Computer Vision - ECCV 2004*, volume 3021 of *Lecture Notes in Computer Science*, chapter 36, pages 469–481. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.
- [25] Laurenz Wiskott, Jean-Marc Fellous, Norbert Krüger, and Christoph Von Der Malsburg. Face recognition by elastic bunch graph matching. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 19:775–779, 1997.
- [26] Ferdinando Samaria and Steve Young. Hmm-based architecture for face identification. *Image and Vision Computing*, 12(8):537 – 543, 1994.
- [27] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, 4th edition, 2008.

- [28] P. Parveen and Bhavani Thuraisingham. Face recognition using multiple classifiers. In *Tools with Artificial Intelligence, 2006. ICTAI '06. 18th IEEE International Conference on*, pages 179–186, 2006.
- [29] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, September 1995.
- [30] U. Raghavendra, P.K. Mahesh, and Anjan Gudigar. A novel face recognition method using pca, lda and support vector machine. In Natarajan Meghanathan, Nabendu Chaki, and Dhinakaran Nagamalai, editors, *Advances in Computer Science and Information Technology. Computer Science and Engineering*, volume 85 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 241–249. Springer Berlin Heidelberg, 2012.
- [31] Paul Viola and Michael Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57:137–154, 2004.
- [32] Gary Bradski and Adrian Kaehler. *Learning OpenCV*. O'Reilly Media Inc., 2008.
- [33] R. Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 1, pages I–900–I–903 vol.1, 2002.
- [34] Nobuyuki Otsu. A Threshold Selection Method from Gray-level Histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979.
- [35] Philipp Wagner. Local binary patterns in opencv. http://www.bytefish.de/blog/local_binary_patterns/, November 2011.
- [36] Philipp Wagner. Machine learning with opencv2. www.bytefish.de, February 2012.
- [37] Anil Jain, Karthik Nandakumar, and Arun Ross. Score normalization in multimodal biometric systems. *Pattern Recogn.*, 38(12):2270–2285, December 2005.
- [38] Face and ocular challenge series. <http://www.nist.gov/itl/iad/ig/focs.cfm>.
- [39] P. Jonathon Phillips, W. Todd Scruggs, Alice J. O'Toole, Patrick J. Flynn, Kevin W. Bowyer, Cathy L. Schott, and Matthew Sharpe. Fvt 2006 and ice 2006 large-scale experimental results. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(5):831–846, May 2010.
- [40] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The det curve in assessment of detection task performance. pages 1895–1898, 1997.



Presupuesto

1) Ejecución Material

▪ Compra de ordenador personal (Software incluido)	1900 €
▪ Alquiler de impresora láser durante 6 meses	200 €
▪ Material de oficina	200 €
▪ Total de ejecución material	2300 €

2) Equipamiento del sistema desarrollado

▪ Ordenador con sistema implementado	800 €
▪ Cámara web	85 €
▪ Televisor	600 €
▪ Soporte para televisor	200 €

3) Gastos generales

▪ sobre Ejecución Material	368 €
----------------------------	-------

4) Beneficio Industrial

▪ sobre Ejecución Material	138 €
----------------------------	-------

5) Honorarios Proyecto

▪ 1800 horas a 15 €/ hora	27000 €
---------------------------	---------

6) Material fungible

▪ Gastos de impresión	150 €
▪ Encuadernación	200 €

7) Subtotal del presupuesto

▪ Subtotal Presupuesto	34141 €
8) I.V.A. aplicable	
▪ 21 % Subtotal Presupuesto	7169,51 €
9) Total presupuesto	
▪ Total Presupuesto	41310,51 €

Madrid, Julio 2013

El Ingeniero Jefe de Proyecto

Fdo.: Javier Eslava Ríos

Ingeniero Superior de Telecomunicación



Pliego de condiciones

Pliego de condiciones

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un *sistema de reconocimiento biométrico basado en la forma de andar*. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales.

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.
2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.
3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.
4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.
6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.
7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.
8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.
9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.
10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.
11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.
12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.
13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.
15. La garantía definitiva será del 4
16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.
17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.
18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.
19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.
20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.
21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.
22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.
23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrataz anteriormente llamado "Presupuesto de Ejecución Material"que hoy designa otro concepto.

Condiciones particulares.

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.
3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.
4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.
5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.
6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.
7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.
8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.
9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.
10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.
11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.
12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.



Anexo C: manuales de utilización.

En esta sección se incluyen los manuales de utilización con instrucciones de uso y consideraciones a tener en cuenta de los dos sistemas off-line y en tiempo real.

C.1. Sistema off-line

El sistema off-line está programado íntegramente en C++ haciendo uso de las librerías de OpenCV. Para compilarlo se requiere que el equipo tenga correctamente configuradas dichas librerías. Los archivos y carpetas en la misma ruta del ejecutable que necesita el programa para su correcto funcionamiento son los siguientes:

- Carpeta *cascades*.
 - *haarcascade_frontalface_alt.xml*. Archivo con las cascadas entrenadas para la detección de cara.
 - *haarcascade_lefteye_2splits.xml*. Archivo con las cascadas entrenadas para la detección del ojo izquierdo.
 - *haarcascade_righteye_2splits.xml*. Archivo con las cascadas entrenadas para la detección del ojo derecho.
- *dev.txt*. Archivo con las rutas completas de las imágenes que serán utilizadas para el entrenamiento del PCA.
- *train.txt*. Contiene las rutas absolutas de las imágenes que serán utilizadas para el entrenamiento.
- *test.txt*. En este archivo se encuentran las rutas absolutas de las imágenes utilizadas como imágenes test.

Los archivos de las imágenes tendrán en el nombre el identificador único de usuario separado por un guión bajo del resto del mismo (ejemplo: *0021_02g1.JPG*). Para las diferentes técnicas,

las funciones del sistema están comentadas en el código y simplemente hay que comentar y descomentar las funciones pertinentes para utilizarlo con unas técnicas u otras.

Al ejecutarse, el programa genera un archivo llamado *experimento.txt* que contiene todos los resultados obtenidos. Este archivo nunca es borrado en ejecuciones posteriores sino que el programa añade los datos al final del mismo. Es por esto por lo que es un funcionamiento a tener en cuenta para no mezclar resultados de diferentes fuentes.

C.2. Sistema en tiempo real

El sistema en tiempo real utiliza además de las librerías de OpenCV, las funciones de Qt. Su utilización puede hacerse desde la plataforma de desarrollo Qt (con las librerías de OpenCV configuradas correctamente) o bien desde el ejecutable ya compilado. Los archivos y carpetas necesarios para el funcionamiento son:

- Carpeta *Cascades*.
 - *haarcascade_frontalface_alt.xml*. Archivo con las cascadas entrenadas para la detección de cara.
 - *haarcascade_lefteye_2splits.xml*. Archivo con las cascadas entrenadas para la detección del ojo izquierdo.
 - *haarcascade_righteye_2splits.xml*. Archivo con las cascadas entrenadas para la detección del ojo derecho.
- Carpeta *DB*. Carpeta donde se almacenan las imágenes entrenadas.

En el caso de la carpeta *DB* los archivos que almacena son imágenes ya normalizadas de cada usuario. Dichos archivos tendrán una nomenclatura parecida a los del sistema off-line (ejemplo *Javier_32.JPG*). Para facilitar la clasificación de los archivos, después del identificador de usuario se puede añadir un número ya que el programa es capaz de leer estos números y seguir la numeración en entrenamientos posteriores.

Como se detalló en el capítulo 3, una vez ejecutado el programa, la interfaz principal se mostrará con tres botones principales para comenzar el funcionamiento. Al pulsar cualquiera de los tres botones, el sistema se inicializará con las técnicas seleccionadas. A partir de este momento los botones del menú que estaban deshabilitados podrán utilizarse. Haciendo click sobre cualquiera de las casillas hará que el sistema se ejecute en el modo seleccionado (identificación, demostración o entrenamiento). Una vez que alguno de los funcionamientos haya comenzado, el programa deshabilitará las casillas del resto de modos hasta que no se desactive ya que el programa no puede funcionar en más de un modo a la vez.

Durante el modo de entrenamiento, el campo de texto servirá para designar el identificador de usuario, no están permitidos los guiones bajos o los espacios puesto que son caracteres de escape para el sistema. Si existe un usuario con el mismo identificador, el programa avisará del conflicto y se podrá elegir cambiar el identificador o añadir las imágenes que se van a entrenar al usuario existente. Si no se selecciona ninguna cantidad de imágenes a entrenar, al pulsar la casilla de entrenamiento el programa comenzará a almacenar imágenes indefinidamente hasta que, volviendo a hacer click sobre la misma casilla, el proceso sea detenido por el usuario.

El programa generará en la carpeta donde se esté ejecutando los siguientes archivos:

- *score_log_pca.txt*. Resultados y decisiones generados por el reconocimiento mediante técnicas de PCA con distancia euclídea.
- *score_log_lbp.txt*. Resultados y decisiones generados por el reconocimiento mediante técnicas de LBP con Chi-Square.
- *score_log_fusion.txt*. Resultados y decisiones generados por el reconocimiento mediante las dos técnicas anteriores fusionadas a nivel de score.
- *time_log.txt*. Tiempos de procesamiento de cada etapa junto con el modo de funcionamiento al que pertenecen.

Estos archivos al igual que en el sistema off-line, no son borrados cada vez que se ejecuta el programa por lo que es importante tener esto en consideración.



Anexo D: manual del programador

En este anexo se detallan las cabeceras de las funciones que utiliza el sistema. Puesto que el sistema en tiempo real utiliza todas las funciones del sistema off-line, las funciones aquí expuestas serán todas las del sistema en tiempo real.

```
1
2
3 Rect* detect_and_draw( IplImage* image, int identified );
4 // Function prototype for detecting and drawing an object from an image
5 // Inputs:   image - Input image
6 //           identified - Flag to know if the detected face has been ←
7 //           identified
8 // Outputs:  Rect - Rectangle of the detected area
9
10 Mat detect_eyes(Mat img, Rect r, int res);
11 // Function prototype for detecting eyes
12 // Inputs:   img - Input image
13 //           r - Rectangle for the previously detected face
14 //           res - Resolution of the image
15 // Outputs:  Mat - Image with eye coordinates drawn.
16
17 Mat rotate_eyes(Mat img, eye reye, eye leye);
18 // Rotate a face image by inputing the eyes coordinates
19 // Inputs:   img - Input image
20 //           reye - Structure containing the coordinates for the right ←
21 //           eye
22 //           leye - Structure containing the coordinates for the left eye
23 // Outputs:  Mat - Rotated image
24
25 Mat scale_and_border_isov(Mat img, eye reye, eye leye, double distance);
26 // Scales an image to ISO standards and adds a black border
27 // Inputs:   img - Input image
28 //           reye - Structure containing the coordinates for the right ←
29 //           eye
```

```

27 //      leye – Structure containing the coordinates for the left eye
28 //      distance – Interpupilar Distance (IPD)
29 // Outputs:  Mat – Scaled image
30
31 void find_peaks (Mat hist, int neighbor=12);
32 // Function that finds the peaks of an histogram
33 // Inputs:    hist – Input histogram
34 //           neighbor – Value to specify the width of the peak
35
36 Mat binarize(Mat img);
37 // Binarize an image with Otsu's method
38 // Inputs:    img – Input image
39 // Outputs:   Mat – Binarized image
40
41 void eyes_coord_y(Mat y_lvector, Mat y_rvector, Rect r, Rect rect_leye, ←
    Rect rect_reye);
42 // Function that finds the y coordinate of the eyes
43 // Inputs:    y_lvector – vector with black values of the left eye summed ←
    horizontally
44 //           y_rvector – vector with black values of the right eye summed ←
    horizontally
45 //           r – Rectangle of the previously detected face
46 //           rect_leye – Rectangle of the cropped area for the left eye
47 //           rect_reye – Rectangle of the cropped area for the right eye
48
49 void eyes_coord_x(Mat x_lvector, Mat x_rvector, Rect r, Rect rect_leye, ←
    Rect rect_reye);
50 // Function that finds the x coordinate of the eyes
51 // Inputs:    x_lvector – Vector with black values of the left eye summed ←
    vertically
52 //           x_rvector – Vector with black values of the right eye summed ←
    vertically
53 //           r – Rectangle of the previously detected face
54 //           rect_leye – Rectangle of the cropped area for the left eye
55 //           rect_reye – Rectangle of the cropped area for the right eye
56
57 Mat* crop_area(Mat m_img, Rect r);
58 // Crops a rectangle of an image
59 // Inputs:    m_img – Input image
60 //           r – Rectangle of the previously detected face
61 // Outputs:   Mat – Cropped image
62
63 Mat sumrows(Mat img_eyes);
64 // Creates a row vector of the summed values of the input image
65 // Inputs:    img_eyes – Input image
66 // Outputs:   Mat – row vector of summed values
67
68 Mat sumcols(Mat img_eyes);
69 // Creates a column vector of the summed values of the input image
70 // Inputs:    img_eyes – Input image
71 // Outputs:   Mat – column vector of summed values
72
73 Mat qimage2mat(const QImage& qimage);
74 // Converts a QImage (Qt image format) to a Mat (OpenCV image format)
75 // Inputs:    qimage – Input image
76 // Outputs:   Mat – Converted image

```



```

77
78 Mat asColSingleMatrix(const Mat src, int rtype);
79 // Transforms a matrix into a column vector
80 // Inputs:   src - Input image
81 //           rtype - Type of the data the matrix holds (int, double, ←
//             float, etc)
82 // Outputs:  Mat - Column vector
83
84 Mat asColMatrix(const vector<Mat>& src, int rtype);
85 // Transforms a vector of matrices into columns to generate a single ←
//   matrix
86 // Inputs:   src - Input vector of images
87 //           rtype - Type of the data the matrix holds (int, double, ←
//             float, etc)
88 // Outputs:  Mat - Resulting matrix
89
90 Mat matrix_multiplication(Mat A, Mat B);
91 // Multiplies two matrices performing (AxB)
92 // Inputs:   A - First input matrix
93 //           B - Second input matrix
94 // Outputs:  Mat - Resulting matrix (AxB)
95
96 CvSVM train_svm(Mat train_data, string label);
97 // Trains the two-class SVM
98 // Inputs:   train_data - Data to train (each class is one row)
99 //           label - Label for one of the classes
100 // Output:   CvSVM - Trained SVM
101
102 void OLBP(const Mat& src, Mat& dst);
103 // Performs a 1-radius 8-neighbors LBP operation
104 // Inputs:   src - Input image
105 //           dst - Resulting image
106
107 void ELBP(const Mat& src, Mat& dst, int radius = 1, int neighbors = 8);
108 // Performs a N-radius M-Neighbours LBP operation
109 // Inputs:   src - Input image
110 //           dst - Resulting image
111 //           radius - radius of the LBP operator
112 //           neighbors - number of samples to obtain per neighborhood
113
114 void VARLBP(const Mat& src, Mat& dst, int radius = 1, int neighbors = 8);
115 // Performs a N-radius M-neighbours Variance Based LBP operation
116 // Inputs:   src - Input image
117 //           dst - Resulting image
118 //           radius - radius of the LBP operator
119 //           neighbors - number of samples to obtain per neighborhood
120
121 void spatial_histogram(const Mat& src, Mat& spatialhist, int numPatterns, ←
//   int gridx=8, int gridy=8, int overlap=0);
122 // Calculates the spatial histogram of an image
123 // Inputs:   src - Input image
124 //           spatialhist - Calculated histogram
125 //           numPatterns - Size of the histogram (Number of bins X number ←
//   of regions)
126 //           gridx - Number of divisions in the x axis
127 //           gridy - Number of divisions on the y axis

```

```

128 //          overlap – Pixels for each region to overlap
129
130 void histogram(const Mat& src, Mat& hist, int numPatterns);
131 // Calculates the histogram of an image
132 // Inputs:   src – Input image
133 //          hist – Calculated histogram
134 //          numPatterns – Size of the histogram (number of bins)
135
136 double chi_square(const Mat& histogram0, const Mat& histogram1);
137 // Performs a Chi-Square calculation between two histograms
138 // Inputs:   histogram0 – Input histogram
139 //          histogram1 – Input histogram
140 // Outputs:  double – Result of the operation
141
142 void captureFromFile( const QString& path );
143 // Attempts to load video file
144 // Inputs: path – Path of the file to load
145
146 void captureFromCamera( int cameraIndex = -1 );
147 // Attempts to start camera,
148 // Inputs: cameraIndex – camera number (see cvCaptureFromCAM docs)
149
150 Mat initialize_PCA();
151 // Initializes and trains the system to function with PCA and Euclidean ↔
    Distance techniques
152
153 void initialize_LBP();
154 // Initializes and trains the system to function with LBP and Chi-Square ↔
    techniques
155
156 void initialize_Fusion();
157 // Initializes and trains the system to function with Fusion techniques
158
159 void master_initialize();
160 // Resets the system in order to clear all variables
161
162 vector<float> euclidean_dist(cv::Mat vect1, cv::Mat vect2);
163 // Performs the Euclidean Distance between two vectors
164 // Inputs:   vect1 – Input vector
165 //          vect2 – Input vector
166
167 void enableProcessingPCA( bool on );
168 // Switch identification with PCA on / off
169 // Inputs:   on – Flag to turn on/off the system
170
171 void enableProcessingLBP(bool on);
172 // Switch identification with LBP on / off
173 // Inputs:   on – Flag to turn on/off the system
174
175 void enableProcessingFusion(bool on);
176 // Switch identification with fusion on / off
177 // Inputs:   on – Flag to turn on/off the system
178
179 void enableTraining( bool on, QString name );
180 // Switch training on / off
181 // Inputs:   on – Flag to turn on/off the system

```

```
182 //          name – label for the training model
183
184 void startDemo(bool on);
185 // Switch Demo mode on / off
186 // Inputs:    on – Flag to turn on/off the system
187
188 void setNumFaces(int value);
189 // Sets the number of faces to detect
190 // Inputs:    value – Index of the dropdown menu in the interface
191
192 void setNumFacesTrain(int value);
193 // Sets the number of faces to train
194 // Inputs:    value – Index of the menu in the interface
195
196 void setResolution(int value);
197 // Sets the resolution to the selected value
198 // Inputs:    value – Index of the dropdown menu in the interface
199
200 void flipImage( bool on );
201 // switch image flipping on / off
202 // Inputs:    on – Flag to turn on/off the flipping system
203
204 void stopCapture();
205 // stops timer and release OpenCV capture structure
```